Claim Amendments

Please cancel all claims of record and substitute the new claims 115 to 157 in Enclosure A.

and distinctly so as to overcome the technical rejections and define the invention patentably over the prior art.

1. This Amendment H is in response to the Office Action mailed September 10, 2003.

The Objection to the Drawings Under 37 CFR 1.83(a)

2. The drawings objections are noted and are corrected with new drawings submitted herewith.

2a. New drawings Fig. 26 shows the feature of "method of determining whether an asynchronous process should or should not be converted into a new periodic process, by calculating whether a ratio of processing capacity of the processor which is required to be reserved for the new periodic process, to a processor capacity that is required for the asynchronous process if left unconverted, exceeds a predetermined threshold value."

2b. New drawing Fig. 25A, Fig. 25B shows the feature of "a permitted range of offset of each new periodic process being a subinterval of an interval or a full interval that begins at the earliest time that the corresponding being converted asynchronous process can make a request for execution, and ends at a time equal to the sum of the earliest time that said being converted asynchronous process can make a request for execution plus the period length of the new periodic process minus one time unit."

2c. New drawings Fig. 27A, Fig. 27B shows the feature of "comprising during run-time, detecting, in a case in which no asynchronous process or periodic process that has started is to be immediately put into execution, conditions of whether there exists an execution of a first periodic process that is ready for execution and has not completed execution, and there does not exist any other execution of some second periodic process that has not yet completed, such that execution of the second periodic process is ordered before execution of the first periodic process in the pre-run-time schedule, and the time slot of the first periodic process is not nested within the time slot of the second periodic process in the pre-run-time schedule, and there does not exist any other execution of some third periodic process that is ready and has not completed execution, such that execution of the third periodic process is nested within the time slot of the first periodic process in the pre-run-time schedule, and beginning execution of the first periodic process immediately in the event said conditions are true."

Applicant hereby declares that the new drawings Fig. 25B, 26, 27A, 27B include no new matter.

Applicant submits that the drawings now comply with 37 CFR 1.83(a) and therefore requests withdrawal of this objection.

The Objection to the Claims Because of Informalities

3. The last O.A. objected to claims 86-88 and 90 because of the spelling error: "processers" should be spelled "processors". Applicant has cancelled claims 86-88 as Applicant considers them to be redundant over other rewritten claims.

4. The last O.A. objected to claim 93 because it was said that "some" should be changed to "a". Applicant has rewritten claim 93 as new claim 139 in which "some" has been changed to "a".

Accordingly Applicant submits that claim 139 is now in proper form and therefore requests withdrawal of this objection.

The Claims Rejection Under 35 USC § 112

5. The last O.A. rejected claim 59 under § 112, since it was said that there was insufficient antecedent basis for the limitation "the step of converting at least one asynchronous process" in the first two lines of the claim. Applicant has cancelled claims 59 as Applicant considers it to be redundant over other rewritten claims.

6. The last O.A. rejected claims 64-65, 74-75, 83-84, 91, 94-95, 99-103, 106, and 112 under § 112, since it was said that the above claims were indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention, since it was said that the term "latitude" is indefinite.

Claims 64-65; 75; 83-84; 91; 94-95; 99-103; and 112 have been replaced with substitute new claims 119-120; 124-125, 129; 134-135; 138; 140-141; 145-149; and 150, respectively. Applicant has cancelled claim 106 as Applicant considers it to be redundant over other rewritten claims.

Applicant requests reconsideration and withdrawal of this objection for the following reasons:

The present specification clearly defines the meaning of the term "latitude", and clearly teaches what specific values one should assign to it, and how to use it in the practice of the invention.

First, paragraph [0127] of the specification teaches that, *"The `latitude' of a process x, denoted by L_x, is a user defined measure of the latitude in meeting process x's deadline."*

Second, paragraph [0128] of the specification teaches that, *"For exemplary purposes, in all the examples given in the description of the present invention, we will assume that for each process x, L_x is set to the following value:*

- *for each periodic process p_i, $L_{pi} = d_{pi} - r_{pi}$;*
- *for each asynchronous process a_i, $L_{ai} = d_{ai}$;*

... ".

(d_{pi}, r_{pi}, d_{ai} above are the deadline and release time of a periodic process p_i, the deadline of an asynchronous process a_i, respectively, as defined in paragraphs [0114] and [0117] of the specification.)

Paragraph [0129] of the specification teaches another example of specific values that one can assign to it, *"... for each P-h-k process or P-s-k process p_i, instead of defining $L_{pi} = d_{pi} - r_{pi}$, $L_{pi} = d_{pi}$ could be defined, or any other criteria for defining L_{pi} could be used."*

Third, whenever the term latitude is used in the claims 119-120, 124-125, 129, 134-135, 138, 140-141, 145-149, and 150, corresponding use of the symbol form of latitude L_{pi}, L_{ai}, etc , is shown in specific paragraphs of the specification related to the claim. For example, claim 119 cites, *"... sufficient time capacity for execution of asynchronous processes that have less latitude than considered ones of periodic processes in meeting their respective deadlines."* Paragraphs [0145]-[0148] of the specification teaches this in detail, *"... Above, for each process p_i in S_P (the original set of P-h-k processes) or in S_P (the new periodic processes converted from A-h-k-p processes), for every possible occurrence of any A-h-k-a process a_j between r_{pi} and d_{pi}, if $L_{aj} < L_{pi}$ then a_j's computation time is added to p_i's computation time."*

Accordingly Applicant submits that the present specification clearly defines the meaning of the term "latitude", and clearly teaches what specific values one should assign to it, and how to use it in the practice of the invention; thus Applicant submits that the claims 119-120, 124-125, 129, 134-135, 138, 140-141, 145-149, and 150 do comply with § 112 and therefore requests withdrawal of this objection.

7. The last O.A. rejected claim 63 under § 112, since it was said that the above claims were indefinite for failing to particularly point out and distinctly claim the subject

matter which applicant regards as the invention, since it was said that the term "satisfied" is indefinite.

Claim 63 has been rewritten as new claim 118. Applicant submits that claim 118 now does comply with § 112 and therefore requests withdrawal of this objection.

8. The last O.A. rejected claims 62-63, 78-80, 104, and 110-111 under § 112, since it was said that the above claims were incomplete for omitting essential structural cooperative relationships. The last O.A. said that it was not clear if "relations" in claims 62-63, 78-80, 104, and 110-111 refers to "exclusion relations" or "relations comprising worst-case computation time."

Claims 62-63, and 78-80, 104, and 110 have been rewritten as new claims 118; 127, 154; 128; 130, 157; 115; 127 respectively. Applicant has cancelled claims 111 as Applicant considers it to be redundant over other rewritten claims. Claims 59, 58 has been rewritten as new claims 115, 116, 153 in which "predetermined constraints" comprises "exclusion relations". Claims 118, and 127, 154, 128, 130, 157 now refer to "said predetermined constraints", which comprises the "exclusion relations" referred to in claim 115 and 153.

Applicant submits that the claims 118; 127, 154; 128; 130, 157; 115 now do comply with § 112 and therefore requests withdrawal of this objection.

9. The last O.A. rejected claims 62-63, and 78-80, 104, and 110 under § 112, since it was said that the above claims were indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention, since it was said that the term "relations" is indefinite.

Claims 62-63, 78-80, 104, and 110 have been rewritten as new claims 118; 127, 154; 128; 130, 157; 115; 127, respectively. Applicant has cancelled claims 111 as Applicant considers it to be redundant over other rewritten claims. Claims 59, 58 has been rewritten as new claims 115, 116, 153 in which "predetermined constraints" comprises "exclusion

relations". Claims 118; 127, 154; 128; 130, 157; 127 now refer to "said predetermined constraints", which comprises the "exclusion relations" referred to in claim 115 and 153.

Applicant submits that the claims 118; 127, 154; 128; 130, 157; 127 now do comply with § 112 and therefore requests withdrawal of this objection.

The Claims Rejection Under 35 USC § 102

10. The Rejection Of Claim 112 Under 35 USC § 102 On Dave (US 6,178,542 B1) Is Overcome

11. The last O.A. rejected independent claim 112 on Dave. Claim 112 has been rewritten as new claim 150 to define patentably over Dave and any combination thereof. Applicant requests reconsideration of this rejection, as now applicable to claim 150, for the same reasons as given in item 13 and item 27 of this amendment, and in addition, the following reasons:

11.1. Dave does not show the feature of satisfying constraints comprising exclusion relations and therefore does not provide the capability to prevent errors caused by more than one periodic process or asynchronous process simultaneously accessing shared resources such as data while maximizing the system's flexibility in meeting deadline constraints

Claim 150 clearly distinguishes over Dave since it recites

"including satisfaction of predetermined constraints comprising

...

(5) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a

periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,"

Dave does not show the feature of satisfying predetermined constraints comprising "*exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,"* and therefore does not provide the capability to prevent errors caused by more than one process simultaneously accessing shared resources such as data while maximizing the system's flexibility in meeting deadline constraints, while the processes can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, (iii) periodic processes that are mapped into time slots in the pre-run-time schedule.

(11.1.1.) Dave's "exclusion vector" (*"This specifies which pairs of tasks cannot co-exist on the same PE (such pairs may create processing bottlenecks)", Dave, col. 6, lines 13-15)* does not show the feature satisfying predetermined constraints comprising exclusion relations hence it is incapable of preventing overlapping in time of the executions of a selected pair of processes. Dave's exclusion vector does not provide the capability to prevent data inconsistencies caused by more than one process simultaneously accessing shared data, because even if a pair of tasks do not coexist on the same PE, they can still simultaneously access shared data and cause data inconsistencies when multiple processors share memory.

(11.1.2) Dave's combination of preemptive and nonpreemptive processing (*"The algorithm employs a combination of both preemptive and nonpreemptive scheduling. Preemptive scheduling is used in restricted scenarios to minimize scheduling complexity." Dave, col.10, lines 1-4).* does not show the feature of satisfying predetermined constraints comprising exclusion relations.

(11.1.2a) Nonpreemptive in general does not provide the capability to prevent errors caused by more than one process simultaneously accessing shared resources such as data when the processes may execute on different processors that share memory, because a pair of tasks that do not coexist on the same PE can still simultaneously access shared data, even if both tasks are executed nonpreemptively.

(11.1.2b) Dave's combination of preemptive and nonpreemptive processing is based on a single parameter called "preemption overhead" *("for task preemption, the algorithm takes into consideration the operating system overheads such as interrupt overhead, context-switch, remote procedure call (RPC) etc. through a parameter called preemption overhead." Dave, col. 10, lines 5-8).* It is not defined on pairs of tasks, thus does not provide the capability to select precisely which pairs of process' executions should not overlap in time when they access shared data. For this reason Dave's combination of preemptive and nonpreemptive processing does NOT offer the same system flexibility in meeting deadline constraints that Applicant's invention as defined by claim 150 provides while Dave also fails to provide the capability to prevent errors caused by more than one process simultaneously accessing shared resources such as data when the processes may execute on different processors that share memory.

(11.1.2c) Dave's combination of preemptive and nonpreemptive processing is incapable of handling processes that can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic

processes that are mapped into time slots in the pre-run-time schedule. Dave does not show any feature related to "*execution of asynchronous processes that are not converted to new periodic processes and hence not mapped to time slots in the pre-run-time schedule in a manner similar to mapping of other periodic processes.*" Dave maps every aperiodic task to specific time slots in the schedule before run-time. *("The algorithm positions execution time slots for aperiodic task graphs throughout the hyperperiod," Dave, col. 7, lines 42-43).*

11.2. Dave does not show the feature of scheduling, between the beginning time and end time of each of the time slots in the pre-run-time schedule reserved for execution of a corresponding periodic process, time capacity sufficient to complete execution of said corresponding periodic process and additional time capacity sufficient to complete execution of asynchronous processes that are not converted to new periodic processes and hence not mapped to time slots in the pre-run-time schedule in a manner similar to mapping of other periodic processes and have less latitude than said corresponding periodic process in meeting their respective deadlines.

Claim 150 clearly distinguishes over Dave since it recites

> *"including scheduling, between the beginning time and end time of each of the time slots in the pre-run-time schedule reserved for execution of a corresponding periodic process, time capacity sufficient to complete execution of said corresponding periodic process and additional time capacity sufficient to complete execution of asynchronous processes that are not converted to new periodic processes and hence not mapped to time slots in the pre-run-time schedule in a manner similar to mapping of other periodic processes and have less latitude than said corresponding periodic process in meeting their respective deadlines."*

The last O.A. cited as reference prior patent 6,178,542 B1 to Dave *("For each aperiodic task, as explained before, the algorithm positions the execution slots throughout the hyperperiod after scheduling the first execution slot. If the execution slot cannot be allocated at the required instant, the algorithm schedules its at the earliest possible time and repositions the remaining slots to ensure that the deadlines are always met." Dave, col. 12, lines 20-26).* Applicant submits that this is a misunderstood reference, as the reference clearly does not teach what the O.A. relies upon it as supposedly teaching.

The reference cited by the O.A. clearly does not show the feature of *"including scheduling, between the beginning time and end time of each of the time slots in the pre-run-time schedule reserved for execution of a corresponding periodic process, time capacity sufficient to complete execution of said corresponding periodic process and additional time capacity sufficient to complete execution of asynchronous processes that are not converted to new periodic processes and hence not mapped to time slots in the pre-run-time schedule in a manner similar to mapping of other periodic processes and have less latitude than said corresponding periodic process in meeting their respective deadlines"* :

11.2.1. Dave does not show any feature related to "*execution of asynchronous processes that are not converted to new periodic processes and hence not mapped to time slots in the pre-run-time schedule in a manner similar to mapping of other periodic processes.*" Dave maps <u>every</u> aperiodic task to specific time slots in the schedule before run-time. *("The algorithm positions execution time slots for aperiodic task graphs throughout the hyperperiod," Dave, col. 7, lines 42-43).*

11.2.2. In Dave, after the arrival of an aperiodic task, that aperiodic task can only be executed in the specific time slots that were previously allocated to it before run-time. This is illustrated in Dave Fig. 2A and Fig. 2B *("Allocation of these two slots* ({2,4} and {6,8} in Fig. 2B) *in the hyperperiod for t2 guarantees that the deadline of t2 is always met ... If t2 arrives before or at instant 2, it will be served by slot {2,4}. If it arrives after instant 2 and before or at instant 6, it will be served by slot {6,8}. Similarly if it arrives*

after instant 6 and before or at instant 12, it will be served by the first slot of the next hyperperiod, and so on. Dave, col. 9, lines 7-14.)

11.3. Dave does not show the feature of during run-time, scheduling of asynchronous processes that are not mapped to time slots combined with scheduling of periodic processes that are mapped to time slots

Claim 150 clearly distinguishes over Dave since it recites:

> *"during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, including allowing executions of asynchronous processes that have not been mapped to time slots in the pre-run-time schedule to be completed within any time slot of a periodic process that has greater latitude in meeting its deadline, such that said predetermined constraints will be satisfied,"*

As Applicant has discussed above, Dave does not show the feature of "*including allowing executions of asynchronous processes that have not been mapped to time slots in the pre-run-time schedule to be completed within any time slot of a periodic process that has greater latitude in meeting its deadline, such that said predetermined constraints will be satisfied,*" Dave maps every aperiodic task to specific time slots in the schedule before run-time. *("The algorithm positions execution time slots for aperiodic task graphs throughout the hyperperiod," Dave, col. 7, lines 42-43).*

11.4. The Novel Features Of Claim 150 Produce New And Unexpected Results And Hence Are Unobvious And Patentable Over Dave Under § 102 and § 103

Claim 150 provides the following combination of features that have never before been provided simultaneously:

(1) Applicant's invention as defined by claim 150 provides the capability to enforce exclusion relations on pairs of processes that can be either (i) asynchronous

processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are mapped into time slots in the pre-run-time schedule, thus providing the capability to prevent errors caused by more than one process simultaneously accessing shared resources such as data in systems of one or more processors while also providing the capability to select precisely which pairs of process' executions should not overlap in time when they access shared data thus maximizing the system's flexibility in meeting deadline constraints.

Dave does not provide this feature.

(2) Applicant's invention as defined by claim 150 simultaneously provides the capability to allow asynchronous processes with very short deadlines, to remain asynchronous and to not be converted to new periodic processes and hence not be allocated specific time slots in the pre-run-time schedule in order to make the most efficient use of processor capacity in a system where all periodic processes are scheduled in a pre-run-time schedule. In Applicant's invention, such asynchronous processes are allowed to execute at run-time within any time slot of a periodic process that has a longer deadline so that such asynchronous with very short deadlines have a much higher chance of meeting their deadlines. In Applicant's invention, this is made possible in part by scheduling additional time for such asynchronous processes in the time slots of periodic processes that have longer deadlines in the pre-run-time schedule.

Dave does not show this feature.

(3) Applicant's invention as defined by claim 150 provides the above combination of features, while providing the capability to handle processes that can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are mapped into time slots in the pre-run-time schedule, thus obtaining the

advantages of pre-run-time scheduling including ability to use the information in the pre-run-time schedule to achieve greater predictability, ability to handle complex constraints, lower run-time overhead, and in general greatly increase the efficiency of scheduling; and the advantages of run-time scheduling including ability to handle asynchronous processes with very short deadlines that cannot be converted into periodic processes or will waste too much processor capacity if converted into periodic processes, while providing a guarantee that all the constraints, including hard deadlines, will be satisfied before run-time.
Dave does not show this combination of features.

A. Applicant's invention as defined by claim 150 achieves **unexpected results**: A system with all the above important features combined together, has never been realized before. The combination of results achieved by Applicant's invention as defined by claim 150 are new and vastly superior compared to that of Dave, or any combination thereof.

B. Applicant's invention as defined by claim 150 is classified in a **crowded art** (the prior art patent Dave cited by the O.A. states that, *"There is a vast amount of literature in the area of scheduling of soft and hard aperiodic tasks", Dave, col. 2, lines 47-48*); therefore, even a small step forward should be regarded as significant.

Applicant therefore submits that claim 150 is patentable under § 102 and § 103 and should be allowed, since they produce new and unexpected results over Dave , or any combination thereof.

The Claims Rejection Under 35 USC § 103

12. The Rejection Of Claim 58 Under 35 USC § 103 On Dave (US 6,178,542 B1) and Dave2 (US 6,086,628) Is Overcome

13. The last O.A. rejected independent claim 58 on Dave and Dave2. Claim 58 has been rewritten as new claim 116 to define patentably over Dave and Dave2 and any combination thereof. Applicant requests reconsideration of this rejection, as now applicable to claim 116, for the following reasons, in addition to the reasons in item 11, item 15, and item 38:

13.1. Neither Dave nor Dave2 show the feature of satisfying constraints comprising exclusion relations and therefore does not provide the capability to prevent errors caused by more than one periodic process or asynchronous process simultaneously accessing shared resources such as data while maximizing the system's flexibility in meeting deadline constraints.

(Discussion on Dave's lack of this feature is provided in item 11.1. above and is equally applicable to Dave and Dave2, and is hereby included here by reference.)

Claim 116 clearly distinguishes over Dave and Dave2 since it recites

"including satisfaction of predetermined constraints comprising

...

(4) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,"

Neither Dave nor Dave2 show the feature of satisfying predetermined constraints comprising "*exclusion relations for periodic and asynchronous processes wherein each*

exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation," and therefore does not provide the capability to prevent errors caused by more than one process simultaneously accessing shared resources such as data while maximizing the system's flexibility in meeting deadline constraints, while the processes can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are mapped into time slots in the pre-run-time schedule.

The last O.A. citation of the features of Dave *("mapping of tasks to processing elements", "finish time", "constraints", col. 1, lines 50-67, and "co-simulation", col. 2, lines 17-43, and "periodic task graphs", "deadlines", col. 4, lines 53-67, and "finish-time estimation step is enhanced by employing a deadline-based scheduling technique", col. 5, lines 1-7, "worst-case execution times", "mapping tasks", col. 5, lines 25-46, "start time", "period", "deadline", col. 5, lines 53-67, "execution time slots", col. 7, lines 40-54),* does not show the Applicant's invention feature of exclusion constraints as cited above.

(Discussion on Dave's lack of this feature is provided in item 11.1. above and is equally applicable to Dave and Dave2, and is hereby included here by reference.)

13.2. Neither Dave nor Dave2 show the feature of permitted range of offset constraints for periodic processes

Claim 116 clearly distinguishes over Dave and Dave2 since it recites

"including satisfaction of predetermined constraints comprising

...

(4) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value, n,"

Neither Dave nor Dave2 show the feature of satisfying predetermined constraints comprising "*permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value, n,"*

The last O.A. cited the following features of Dave2 (*"An association array has an entry for each task of each copy of the task graph and contains information such as: 1) the PE to which it is allocated, 2) its priority level, 3) its deadline, 4) its best-case projected finish time (PFT), and its 5) its worst-case PFT. The deadline of the nth instance of a task is offset by (n-1) multiplied by its period from the deadline in the original task. The association array not only eliminates the need to replicate the task graphs, but it also allows allocation of different task graphs copies to different PEs, if desirable to derive en efficient architecture. This array also supports pipelining of task graphs, which is explained later in this specification.", col. 10, lines 1-12*). The last O.A. said that the above reference teaches using any offset value in a permitted range of offsets. Applicant submits that the above reference is a misunderstood reference, as the reference does not teach what the O.A. relies upon it as supposedly teaching.

Note that:

13.2.1. The term "offset" in the reference above refers to the duration of the time interval between the deadline of the nth instance in the nth period of the same task and the deadline of the first instance in the first period of the task.

13.2.2. In contrast, in Applicant's invention as defined by claim 116, the term "offset" refers to *"the duration of the time interval between the beginning of the first period of said periodic process and time zero" (Constraint (4) of Applicant's claim 116).*

13.2.3. The meaning of the term "offset" in Dave2 is clearly completely different from the meaning of the term "offset" in Applicant's invention as defined by claim 116. Thus the use of the term "offset" in Dave2 does not provide evidence that Dave2 shows the feature of "permitted range of offsets for periodic processes" in claim 116.

13.4. Neither Dave nor Dave2 show the feature in claim 116 of scheduling on one or more processors, executions of a plurality of periodic and asynchronous processes comprising during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, including allowing executions of asynchronous processes that have not been mapped to time slots in the pre-run-time schedule to be completed within any time slot of a periodic process that has greater latitude in meeting its deadline, such that the predetermined constraints will be satisfied, where the predetermined constraints include permitted range of offset constraints and exclusion relations.

Claim 116 clearly distinguishes over Dave and Dave2 since it recites:

"scheduling on one or more processors, executions of a plurality of periodic and asynchronous processes, comprising:

(A)

automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(1) worst-case computation times for periodic processes and asynchronous processes,

(2) period for periodic processes,

(3) minimum time between two consecutive requests for asynchronous processes, deadline for periodic processes and asynchronous processes,

(4) deadline for periodic processes and asynchronous processes,

(5) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(6) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots,

(B)

during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to

schedule the process executions, such that said predetermined constraints will be satisfied.

Neither Dave nor Dave2 show any feature related to *"scheduling on one or more processors, executions of a plurality of periodic and asynchronous processes, comprising during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such that said predetermined constraints will be satisfied", "said predetermined constraints comprising: ... permitted range of offset constraints ..., exclusion relations...."*

The last O.A. cited from Dave: *("scheduling", "tasks", "execution", "start", "finish", col. 9, lines 65-67)* as evidence that Dave shows the feature "(B) during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such that said predetermined constraints will be satisfied." Applicant submits that this is a misunderstood reference, as the reference clearly does not teach what the O.A. relies upon it as supposedly teaching.

Note that the phrases cited by the last O.A. are within the context of Section 4 The CASPER Algorithm in Dave: *("4 The CASPER Algorithm This section provides an overview of CASPER. FIG. 4 presents one possible co-synthesis process flow for the present invention. This flow is divided up into two-parts: pre-processing and synthesis. ... The synthesis step determines the allocation for both periodic and aperiodic graphs. The synthesis has two loops: 1) an outer loop for allocating each cluster, and an inner loop for evaluating various allocations for each cluster. For each cluster, an allocation array consisting of the possible allocations at that step is created. ... The next step is scheduling which determines the relative ordering of tasks/edges for execution and the start time and finish times for each task and edge. The algorithm employs a combination of both preemptive and non-preemptive static scheduling. Preemptive scheduling is used in restricted scenarios to minimize scheduling complexity (See Section 4.4.) For task*

preemption, the algorithm takes into consideration the operating system overheads such as interrupt overhead, context switch, remote procedure call (RPC) etc. through a parameter called preemption overhead (this information is experimentally determined and provided a priori). Incorporating scheduling into the inner loop facilitates accurate performance evaluation. Performance evaluation of an allocation is extremely important in picking the best allocation. An important step of performance evaluation is finish-time estimation. In this step, with the help of the scheduler, the finish times of each task and edge are estimated using the longest path algorithm. See Reference (2). After finish-time estimation, it is verified whether the given deadlines in the task graphs are met. The allocation evaluation step compares the current allocation against previous ones based on total dollar cost of the architecture." Dave, col. 9, lines 32-67, to col. 10, lines 1- 20).

Note that:

(1) All the phrases cited by the last O.A. *("scheduling", "tasks", "execution", "start", "finish", col. 9, lines 65-67)* are part of the Section 4 of Dave above that describe the "CASPER *(Co-synthesis of Aperiodic SPecification of Embedded system aRchitectures, Dave, col. 3, lines 56-57.)"* algorithm, which as its name describes, is a co-synthesis algorithm.

(2) Co-synthesis is a procedure of partitioning a system specification into hardware and software modules, which is notoriously well-known as a procedure that is done off-line, and NOT a procedure that actually executes tasks on a processor at run-time. *("Hardware-software co-synthesis is the process of partitioning an embedded system specification into hardware and software modules to meet performance, power, and cost goals." Dave, Abstract, lines 1- 4.)*

(3) The activity of "scheduling" *("scheduling", "tasks", "execution", "start", "finish", col. 9, lines 65-67)* cited by the last O.A. is part of the "inner loop" of the CASPER co-synthesis algorithm (*"Incorporating scheduling into the inner loop facilitates accurate performance evaluation" Dave, col. 10, lines 10-11*) and is only used to "evaluate" various allocations for each cluster *("The synthesis has two loops: 1) an outer loop for allocating each cluster, and an inner loop for evaluating various allocations for each cluster". Dave, col. 9, lines 55-58).*, so it

is also part of co-synthesis that is done off-line, and it is NOT an activity that actually executes tasks on a processor at run-time.

(4) Fig. 4 (Sheet 3 of Dave), which shows CASPER, also shows that the activities associated with the phrases cited by the last O.A. *("scheduling", "tasks", "execution", "start", "finish", col. 9, lines 65-67),* which are part of "PERFORM SCHEDULING AND FINISH-TIME ESTIMATION" in the " inner loop" of Fig. 4, also clearly indicate that these activities are only part of an off-line algorithm to partition a system specification into a "FINAL SOLUTION", and do NOT actually executes tasks on a processor at run-time. (This is shown even more clearly in the box labled "SCHEDULING" in the "inner loop" of Fig. 3 of Dave2, as explained in (8) below.)

Similar to Dave, Dave2 is also a co-synthesis algorithm that is performed off-line and does not actually execute tasks on a processor at run-time:

Note that:

(5) All the phrases cited by the last O.A. *("scheduling", "tasks", "execution", "start", "finish", Dave, col. 9, lines 65-67)* are also part of the Section 2 of Dave2 that describe the COSYN algorithm *(The present invention is related to a heuristic-based co-synthesis technique, called COSYN, which includes allocation, scheduling, and performance estimation steps as well as power optimization features, Dave2, col. 2, lines 49-52.)" ("The next step is scheduling which determines the relative ordering of tasks/edges for execution and the start time and finish times for each task and edge."Dave2, col. 8, lines 56-58.)*, which as its name describes, is a co-synthesis algorithm.

(6) Co-synthesis is a procedure of partitioning a system specification into hardware and software modules, which is notoriously well-known as a procedure that is done off-line, and NOT a procedure that actually executes tasks on a processor at run-time. *("Hardware-software co-synthesis is the process of partitioning an embedded system specification into hardware and software modules to meet performance, power, and cost goals." Dave2, Abstract, lines 1- 4.)*

(7) The activity of "scheduling" *("scheduling", "tasks", "execution", "start", "finish", Dave, col. 9, lines 65-67)* cited by the last O.A. is also part of the "inner loop" of the COSYN co-synthesis algorithm of Dave2 (*"Incorporating scheduling into the inner loop facilitates accurate performance evaluation" Dave2, col. 8, lines 64-66*) and is only used to "evaluate" various allocations for each cluster (*"The synthesis has two loops: 1) an outer loop for allocating each cluster, and an inner loop for evaluating various allocations for each cluster". Dave2, col. 8, lines 41-43).*, so it is also part of co-synthesis that is done off-line, and it is NOT an activity that actually executes tasks on a processor at run-time.

(8) Fig. 3 (Sheet 3 of Dave2), which shows COSYN *("2 The COSYN Algorithm In this section, an overview of the COSYN algorithm is provided followed up by details on each ste. FIG. 3 presents a co-synthesis process flow, according to one embodiment of the present invention." Dave2, col. 8, lines 14-18.)* also shows that the activities associated with the phrases cited by the last O.A. *("scheduling", "tasks", "execution", "start", "finish", Dave, col. 9, lines 65-67), ("The next step is scheduling which determines the relative ordering of tasks/edges for execution and the start time and finish times for each task and edge."Dave2, col. 8, lines 56-58.)* which is clearly shown as the box labeled "SCHEDULING" in the " inner loop" of Fig. 3, also clearly indicate that these activities are only part of an off-line algorithm to partition a system specification into a "FINAL SOLUTION" (Fig. 3), and do NOT actually executes tasks on a processor at run-time.

The above clearly and unequivocally shows that the references Dave and Dave2 cited by the O.A. do NOT show the feature in claim 116 of "(B) during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such that said predetermined constraints will be satisfied."

13.4(a) Claim 116 recites during run-time using the information in the pre-run-time schedule to schedule the process executions, which include periodic processes and asynchronous processes. Neither Dave nor Dave2 show during run-time scheduling periodic processes together with asynchronous processes. In Dave and Dave 2, both synthesis and scheduling is performed off-line as explained in (1)-(4) above.

13.4(b) Claim 116 recites during run-time using the information in the pre-run-time schedule to schedule the process executions, which include periodic processes and asynchronous processes, such that said predetermined constraints will be satisfied.
Dave does not show during run-time using the information in the pre-run-time schedule to schedule the process executions, which include periodic processes and asynchronous processes such that said predetermined constraints will be satisfied. In Dave and Dave2, both synthesis and scheduling is performed off-line, so no constraints are considered at run-time.

13.4(c) Claim 116 recites during run-time scheduling of periodic processes and asynchronous processes, such that said predetermined constraints will be satisfied, where the predetermined constraints include permitted range of offset constraints and exclusion relations.
Neither Dave nor Dave2 show during run-time scheduling of periodic processes and asynchronous processes such that said predetermined constraints will be satisfied where the predetermined constraints include permitted range of offset constraints and exclusion relations. Neither Dave nor Dave2 consider permitted range of offset constraints and exclusion relations as defined in claim 116.

13.5. The Novel Features Of Claim 116 Produce New And Unexpected Results And Hence Are Unobvious And Patentable Over Dave and Dave 2 Under § 103

Claim 116 provides the following combination of features that have never before been provided simultaneously:

(1) Applicant's invention as defined by claim 116 provides the capability to enforce exclusion relations on pairs of processes that can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are mapped into time slots in the pre-run-time schedule, thus providing the capability to prevent errors caused by more than one process simultaneously accessing shared resources such as data in systems of one or more processors while also providing the capability to select precisely which pairs of process' executions should not overlap in time when they access shared data thus maximizing the system's flexibility in meeting deadline constraints.

Neither Dave nor Dave2 show this feature.

(2) Applicant's invention as defined by claim 116 simultaneously provides the capability to increase the flexibility of meeting deadline constraints when there is flexibility in assigning an offset value for a periodic process.

Neither Dave nor Dave2 show this feature.

(3) Applicant's invention as defined by claim 116 provides the above combination of features, while providing the capability to handle processes that can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are mapped into time slots in the pre-run-time schedule, thus obtaining the advantages of pre-run-time scheduling including ability to use the information in the pre-run-time schedule to achieve greater predictability, ability to handle complex constraints, lower run-time overhead, and in general greatly increase the efficiency of scheduling; and the advantages of run-time scheduling including ability to handle asynchronous processes with very short deadlines that cannot be converted into periodic processes or will waste too much processor capacity if

converted into periodic processes, while providing a guarantee that all the constraints, including hard deadlines, will be satisfied before run-time. Neither Dave nor Dave2 do not show this combination of features.

A. Applicant's invention achieves **unexpected results**: A system with all the above important features combined together, has never been realized before. The combination of results achieved by Applicant's invention are new and vastly superior compared to that of Dave and Dave2, or any combination thereof.

B. Applicant's invention is classified in a **crowded art** (a prior art patent cited by the O.A. states that, *"There is a vast amount of literature in the area of scheduling of soft and hard aperiodic tasks", Dave, col. 2, lines 47-48*); therefore, even a small step forward should be regarded as significant.

Applicant therefore submits that claim 116 is patentable under § 102 and § 103 and should be allowed, since they produce new and unexpected results over Dave and Dave2, or any combination thereof.

14. The Rejection Of Claims 59-65, 68, 75-76, 78-79, 81, 83-85, 89, 91, 95, 86-88, 90, 92, 94, 97-99, 103-110, 113-114 Under 35 USC § 103 On Dave (US 6,178,542 B1), Dave2 (US 6,086,628), and Lindsley (US 6,430,593 B1)

15. The Rejection Of Claim 59 Under 35 USC § 103 On Dave (US 6,178,542 B1), Dave2 (US 6,086,628), and Lindsley (US 6,430,593 B1) Is Overcome

The last O.A. rejected dependent claim 59 on Dave, Dave2, and Lindsley. Claims 59, 104, 105 have been rewritten as new independent claim 115 to define patentably over Dave, Dave2, and Lindsley and any combination thereof. Applicant requests reconsideration of this rejection, as now applicable to claim 115, for the same reasons as given in item 13, item 38 and item 39 of this amendment, and in addition, the following reasons:

(1) There is no justification, in Dave, Dave2, and Lindsley, or in any other prior art separate from Applicant's disclosure, which suggests that these references be combined, much less combined in the manner proposed.

(2) The proposed combination would not be physically possible or operative.

(3) Even if Dave, Dave2, and Lindsley were to be combined in the manner proposed, the proposed combination would not show all of the novel physical features of claim 115.

(4) The novel features of claim 115 produce new and unexpected results and hence are unobvious and patentable over these references.

15.1. Dave, Dave2, and Lindsley Do Not Contain Any Justification To Support Their Combination, Much Less In The Manner Proposed

With regard to the proposed combination of Dave, Dave2, and Lindsley, it is well known that in order for any prior-art references themselves to be validly combined for use in a prior-art § 103 rejection, *the references themselves* (or some other prior art) must suggest that they be combined. E.g., as was stated in In re Sernaker, 217 U.S.P.Q. 1, 6 (C.A.F.C. 1983):

> "[P]rior art references in combination do not make an invention obvious unless something in the prior art references would suggest the advantage to be derived from combining their teachings."

That the suggestion to combine the references should not come from applicant was forcefully stated in Orthopedic Equipment Co. v. United States, 217 U.S.PQ. 193, 199 (CAFC 1983):

> "It is wrong to use the patent in suit [here the patent application] as a guide through the maze of prior art references, combining the right references in the right way to achieve the result of the claims in suit [here the claims pending].

> Monday morning quarterbacking is quite proper when resolving the question of nonobviousness in a court of law [here the PTO]."

As was further stated in Uniroyal, Inc. v. Rudkin-Wiley Corp., 5 U.S.P.Q.2d 1434 (C.A.F.C. 1988), "[w]here prior-art references require selective combination by the court to render obvious a subsequent invention, there must be some reason for the combination other than the hindsight gleaned from the invention itself. . . . *Something in the prior art must suggest the desirability and thus the obviousness of making the combination.*" [Emphasis supplied.]

In line with these decisions, the Board stated in Ex Parte Levengood, 28 U.S.P.Q.2d 1300 [P.T.O.B.A.&]. 1993):

"In order to establish a *prima facie* case of obviousness, it is necessary for the examiner to present evidence, preferably in the form of some teaching, suggestion, incentive or inference in the applied prior art, or in the form of generally available knowledge, that one having ordinary skill in the art *would have been led to* combine the relevant teachings of the applied references in the proposed manner to arrive at the claimed invention. . . . That which is within the capabilities of one skilled in the art is not synonymous with obviousness, ... That one can *reconstruct* and/or explain the theoretical mechanism of an invention by means of logic and sound scientific reasoning does not afford the basis for an obviousness conclusion unless that logic and reasoning also supplies sufficient impetus to have led one of ordinary skill in the art to combine the teachings of the references to make the claimed invention. . . . Our reviewing courts have often advised the Patent and Trademark Office that it can satisfy the burden of establishing a *prima facie* case of obviousness only be showing some objective teaching in either the prior art, or knowledge generally available to one of ordinary skill in the art, that 'would lead' that individual 'to combine the relevant teachings of the references." ... Accordingly, an examiner cannot establish obviousness by locating references which describe various aspects of a patent application's invention without also providing

evidence of the motivating force which would impel one skilled in the art to do what the patent applicant has done."

In the present case, there is no reason given in the last O.A. to support the proposed combination, other than the statement *"Referring to claim 59, Dave inherently teaches mapping onto timeslots but fails to explicitly teach a method as defined in claim 58, including the step of converting at least one asynchronous process to a corresponding new periodic process prior to the mapping step, and mapping the new periodic process in a manner similar to mapping of other periodic processes. It is common knowledge in the art of task management that converting an asynchronous process to a new periodic process (or a synchronous process) is known as synchronization. Lindsley teaches a real-time task scheduling system which synchronizes tasks/processes ("processes", synchronized", col. 2, lines 48-61, "asynchronous", col. 4, lines 19-23). It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of synchronization for the reason of increasing speed and efficiency because synchronization is necessary for parallel processing."*

However, the fact that the words "processes", "synchronized" (Lindsley, col. 2, lines 46-61), "asynchronous" (Lindsley, col. 4, lines 19-23) appear in the reference is not sufficient to gratuitously and selectively combine parts of one reference (Lindsley's task synchronization mechanism) with another reference in order to meet Applicant's novel claimed combination of features.

Note that:

15.1.1. The meaning of the term "synchronous" in Lindsley is inherently different from the meaning of "periodic" in Applicant's invention as defined by claim 115 as shown below:

15.1.1.(a) Lindsley uses the term "synchronous" when describing "synchronous task commands". *("The TSA accepts commands from tasks called 'synchronous' task commands. These commands are synchronous from the point of view that the task may not issue another synchronous task command until the previous synchronous task*

command has been completed. It is allowable for the task to perform other activity after issuing a synchronous task command as long as the task verifies that the previous task has been completed prior to issuing another synchronous task command." Lindsley, col. 6, lines 47-55.) It is clear that in Lindsley there is no periodic constraint on "synchronous" task commands, i.e., synchronous task commands are not constrained to execute strictly once in each fixed period of time.

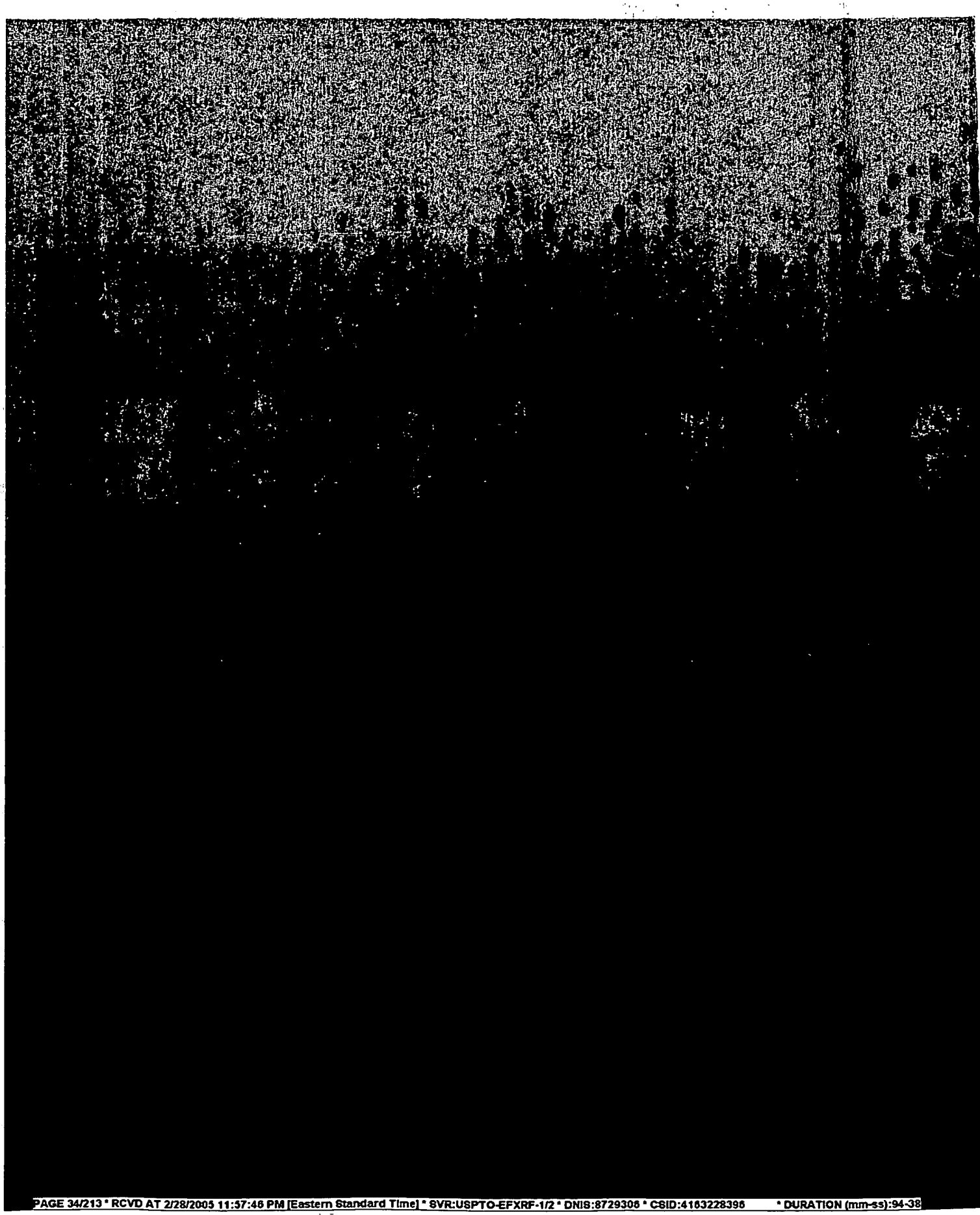
15.1.1.(b) In contrast, in Applicant's invention as defined by claim 115, "a periodic process consists of a computation that is executed repeatedly, once in each fixed period of time. (Applicant's specification, paragraph [0113]). Applicant's definition of a periodic process, is adopted universally in the field of real-time computing and is clearly different from the meaning of "synchronous" as defined in Lindsley.

15.1.2. The meaning of the term "*synchronization*" in Lindsley is also inherently different from the meaning of "*converting an asynchronous process to a new periodic process*" in Applicant's invention as defined by claim 115 as shown below:

15.1.2.(a) In Lindsley, tasks are synchronized using "events" which are certainly NOT always periodic.*("Tasks are typically synchronized using "events". An event is used to indicate that an activity has taken place, such as data arrival, time-out, etc. Thus, an event may indicate execution of a task, an interrupt service routine or the like. Events are counted using semaphores. Semaphores synchronize the event producer and the event consumer ... " Lindsley, col. 2, lines 4-10.)*

("If a task that processes data buffers pends for the semaphore that represents data buffers, the task is synchronized to the data buffer generation. If data buffers are available, the semaphore count is greater than zero. Task pend requests on the semaphore allow the task to continue. If data buffers are not available, the semaphore count is less than or equal to zero, the task does not have data for processing and will block execution". Lindsley, col. 2, lines 52-61.)

15.1.2.(b) In contrast, in Applicant's invention as defined by claim 115, "*converting an asynchronous process to a new periodic process*" means converting a process that can



make a request at random times, to a new process that will be executed once in each fixed period of time while satisfying all the original process' timing constraints.

The above evidence, clearly shows that the last O.A.'s reason for combining Dave, Dave2, and Lindsley to find obviousness of claim 115, i.e., *("It is common knowledge in the art of task management that converting an asynchronous process to a new periodic process (or a synchronous process) is known as synchronization,"* is totally unfounded.

Applicant therefore submits that combining Dave, Dave2, and Lindsley is not legally justified and is therefore improper. Thus Applicant submits that the rejection on these references is also improper and should be withdrawn.

Applicant respectfully requests, if the claims are again rejected upon any combination of references, that the Examiner include an explanation, in accordance with M.P.E.P. § 706.02, Ex parte Clapp, 27 U.S.P.Q. 972 (P.O.B.A. 1985), and Ex parte Levengood, supra, a "factual basis to support his conclusion that it would have been obvious" to make the combination,

15.2. Even if Dave, Dave2, and Lindsley Were To Be Combined In The Manner Proposed, The Proposed Combination Would Not Show All The Novel Features Of Claim 115.

However even if the combination of Dave, Dave2, and Lindsley were justified, claim 115 would still have novel and unobvious features over the proposed combination.

15.2.1. Neither Dave, nor Dave2, nor Lindsley, nor any possible combination thereof show the feature of automatically generating a pre-run-time schedule in which predetermined constraints comprising worst-case computation time, period,

minimum time between two consecutive exclusion relations, deadline, permitted range of offset constraints, precedence relations are satisfied.

(Discussion on Dave's lack of ability to satisfy exclusion constraints is provided in item 11.1. above and is equally applicable to Dave and Dave2 and is hereby included here by reference.)

Claim 115 clearly distinguishes over Dave, Dave2 and Lindsley, or any possible combination thereof, since it recites

"automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(1) worst-case computation times for periodic processes and asynchronous processes,

(2) period for periodic processes,

(3) minimum time between two consecutive requests for asynchronous processes,

(4) deadline for periodic processes and asynchronous processes,

(5) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(6) precedence relations for periodic processes wherein each precedence relation being defined between a pair of processes comprising a first

process and a second process, both said first process and said second process being periodic processes, said first process precedes said second process, execution of said second process only allowed to start after said first process has completed its execution,

(7) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots, including the step of converting one or more asynchronous processes into corresponding new periodic processes prior to the mapping step, and mapping new periodic processes to time slots in a manner similar to mapping of other periodic processes, such that said predetermined constraints will be satisfied".

Neither Dave, nor Dave2, nor Lindsley, nor any possible combination thereof, show the feature of *automatically generating a pre-run-time schedule including satisfaction of predetermined constraints comprising*

"exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation".

(Discussion on Dave's lack of this feature is provided in item 11.1. above and is equally applicable to Dave and Dave2, and is hereby included here by reference.)

Note that Lindsley performs synchronization only at run-time, NOT before run-time, so it is not capable of satisfying any constraints before run-time, which Applicant's invention, as defined by claim 115 has shown.

15.2.2. Neither Dave, nor Dave2, nor Lindsley, nor any possible combination thereof show the feature of permitted range of offset constraints for periodic processes

Claim 115 clearly distinguishes over any combination of Dave, Dave2 and Lindsley since it recites

> *"including satisfaction of predetermined constraints comprising*
>
> ...
>
> *(4) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value, n,"*

Dave, nor Dave2, nor Lindsley, nor any possible combination thereof show the feature of satisfying predetermined constraints comprising "*permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value, n,*"

15.2.3 Dave does not show the feature of scheduling on one or more processors, executions of a plurality of periodic and asynchronous processes comprising during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, including allowing executions of asynchronous processes that have not been mapped to time slots in the pre-run-time schedule to be completed within any time slot of a periodic process that has greater latitude in meeting its deadline, such that the predetermined constraints will be satisfied, where the predetermined constraints include exclusion relations.

Claim 115 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:

"*scheduling on one or more processors, executions of a plurality of periodic and asynchronous processes, comprising:*

(A)

automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(8) worst-case computation times for periodic processes and asynchronous processes,

(9) period for periodic processes,

(10) minimum time between two consecutive requests for asynchronous processes,

(11) deadline for periodic processes and asynchronous processes,

(12) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(13) precedence relations for periodic processes wherein each precedence relation being defined between a pair of processes comprising a first process and a second process, both said first process and said second process being periodic processes, said first process precedes said second process, execution of said second process only allowed to start after said first process has completed its execution,

(14) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots, including the step of converting one or more asynchronous processes into corresponding new periodic processes prior to the mapping step, and mapping new periodic processes to time slots in a manner similar to mapping of other periodic processes, such that said predetermined constraints will be satisfied

(B)

during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such that said predetermined constraints will be satisfied."

Neither Dave, nor Dave2, nor Lindsley show *"during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such that said predetermined constraints will be satisfied."*

The last O.A. stated that, "*Referring to claim 59, Dave inherently teaches mapping onto timeslots but fails to explicitly teach a method as defined in claim 58, including the step of converting at least one asynchronous process to a corresponding new periodic process prior to the mapping step, and mapping the new periodic process in a manner similar to mapping of other periodic processes. It is common knowledge in the art of task management that converting an asynchronous process to a new periodic process (or a synchronous process) is known as synchronization. Lindsley teaches a real-time task scheduling system which synchronizes tasks/processes ("processes", synchronized", col. 2, lines 48-61, "asynchronous", col. 4, lines 19-23). It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of synchronization for the reason of increasing speed and efficiency because synchronization is necessary for parallel processing.*" Applicant respectfully disagrees with the above statement in the last O.A. for the following reasons:

15.2.3.1. As explained in item 15.1.1.(a)-(b), 15.1.2.(a)-(b), Lindsley does not show "*converting one or more asynchronous processes into corresponding new periodic processes prior to the mapping step, and mapping new periodic processes to time slots in a manner similar to mapping of other periodic processes, such that said predetermined constraints will be satisfied*", as the meaning of "synchronous" in Lindsley is totally different from "periodic", and the meaning of "synchronization" in Lindsley is totally different from "converting one or more asynchronous processes into corresponding new periodic processes" hence Lindsley does not show periodic processes at all, and consequently cannot show the above feature.

15.2.3.2. As also explained in item 15.2., Lindsley performs synchronization only at run-time, NOT before run-time, so it is not capable of satisfying any constraints before run-time, which Applicant's invention, as defined by claim 115 has shown.

15.2.3.3. As also explained in item 15.3, Lindsley describes a method which is totally void of any consideration of quantitative timing constraints.
Lindsley does not take into account:

(a) worst-case computation time values of asynchronous or periodic processes,
(b) deadline values of asynchronous or periodic processes,
(c) period values of periodic processes
(d) minimum time between two requests of asynchronous processes
(e) permitted range of offset values for periodic processes.

Hence, Lindsley does not teach "*during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such that said predetermined constraints will be satisfied.*"

15.2.3.4. A explained in item 13.3, (1)-(8) above, both Dave and Dave2 are co-synthesis algorithms that are performed off-line and do NOT actually execute tasks on a processor at run-time.

15.2.3.4(a) Claim 115 recites during run-time using the information in the pre-run-time schedule to schedule the process executions, which include periodic processes and asynchronous processes. Neither Dave nor Dave2 show during run-time scheduling periodic processes together with asynchronous processes. In Dave and Dave 2, both synthesis and scheduling is performed off-line as explained in (1)-(4) above.

15.2.3.4.b) Claim 115 recites during run-time using the information in the pre-run-time schedule to schedule the process executions, which include periodic processes and asynchronous processes, such that said predetermined constraints will be satisfied.

Dave does not show during run-time using the information in the pre-run-time schedule to schedule the process executions, which include periodic processes and asynchronous processes such that said predetermined constraints will be satisfied. In Dave and Dave2, both synthesis and scheduling is performed off-line, so no constraints are considered at run-time.

15.2.3.4.(c) Claim 115 recites during run-time scheduling of periodic processes and asynchronous processes, such that said predetermined constraints will be satisfied, where the predetermined constraints include permitted range of offset constraints and exclusion relations.

Neither Dave nor Dave2 show during run-time scheduling of periodic processes and asynchronous processes such that said predetermined constraints will be satisfied where the predetermined constraints include permitted range of offset constraints and exclusion relations. Neither Dave nor Dave2 consider permitted range of offset constraints and exclusion relations as defined in claim 115. The latter has been explained earlier in item 11.1. and item 15.2.2. respectively.

15.3. The Suggested Combination Of Dave, Dave2, And Lindsley Would Not Be Physically Possible Or Operative.

There is plenty of unequivocal evidence that show that the suggested combination of Dave, Dave2, and Lindsley would not be physically possible or operative, here is just a few of them:

15.3.1. Lindsley describes a method which is totally void of any consideration of quantitative timing constraints.

Lindsley does not take into account:

(f) worst-case computation time values of asynchronous or periodic processes,

(g) deadline values of asynchronous or periodic processes,

(h) period values of periodic processes

(i) minimum time between two requests of asynchronous processes

(j) permitted range of offset values for periodic processes.

15.3.2. It is notoriously well-known, and common sense would also dictate that, if any single one of the above quantitative timing constraints is not taken into consideration, then there is absolutely no hope of satisfying the timing constraints.

15.3.3. The references of Dave and Dave2 do not contain any teaching, that suggests that their method could be physically combined with the teachings of Lindsley. Neither does the reference of Lindsley contain any teaching, that suggests the teaching of Lindsley could be physically combined with the teachings of Dave and Dave2.

15.3.4. There is no evidence in the prior art that suggests that the references of Dave, Dave2, and Lindsley could be combined to produce an operative method.

15.3.5. Combining Dave, Dave2, and Lindsley will NOT produce an operative method that will cover all the features of claim 115. Lindsley does not show any feature that relate to time slots, and Lindsley does not concern itself with satisfying timing constraints, thus it NOT be able to schedule tasks that have been pre-allocated to time slots together with tasks that are not allocated to time slots, while simultaneously satisfying all the timing constraints.

15.3.6. As shown in item 15.1.1 above, Lindsley does not even consider periodic processes.

15.3.7. As shown in item 15.1.1 above, the last O.A.'s citation of Lindsley is a misunderstood reference. The reference does not teach what the last O.A. relies

upon it as supposedly teaching. The main task feature of "synchronous" is not identical to "periodic" as the last O.A. assumed.

15.3.8. The reference of Lindsley relies on the use of general semaphores. It is notoriously well known that general semaphores are not suitable for use in systems with hard deadlines and may prevent the satisfaction of hard deadline constraints altogether. Here is just one simple example: general semaphores are subject to deadlock and various forms of starvation, and just detecting deadlocks alone is an NP-Hard problem, not to mention the total breakdown in any capability to meet any kind of timing constraints when a deadlock does occur. The entire specification of Lindsley completely ignores this problem. As to be expected, the references of Dave and Dave2 also completely ignores this problem.

15.3.9. Even if Dave and Dave2 were operative when functioning alone, and even if it is structurally possible to combine them with Lindsley, the resulting method will be inoperative for the purpose of satisfying hard timing constraints at the very least because of the possibility of deadlock in Lindsley's method.

15.4. The Novel Features Of Claim 115 Produce New And Unexpected Results And Hence Are Unobvious And Patentable Over Any Possible Combination Of Dave, Dave 2 And Lindsley Under § 103

Claim 115 provides the following combination of features that have never before been provided simultaneously:

(1) Applicant's invention as defined by claim 115 provides the capability to enforce exclusion relations on pairs of processes that can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are

mapped into time slots in the pre-run-time schedule, thus providing the capability to prevent errors caused by more than one process simultaneously accessing shared resources such as data in systems of one or more processors while also providing the capability to select precisely which pairs of process' executions should not overlap in time when they access shared data thus maximizing the system's flexibility in meeting deadline constraints.

Neither Dave, nor Dave 2, nor Lindsley show this feature.

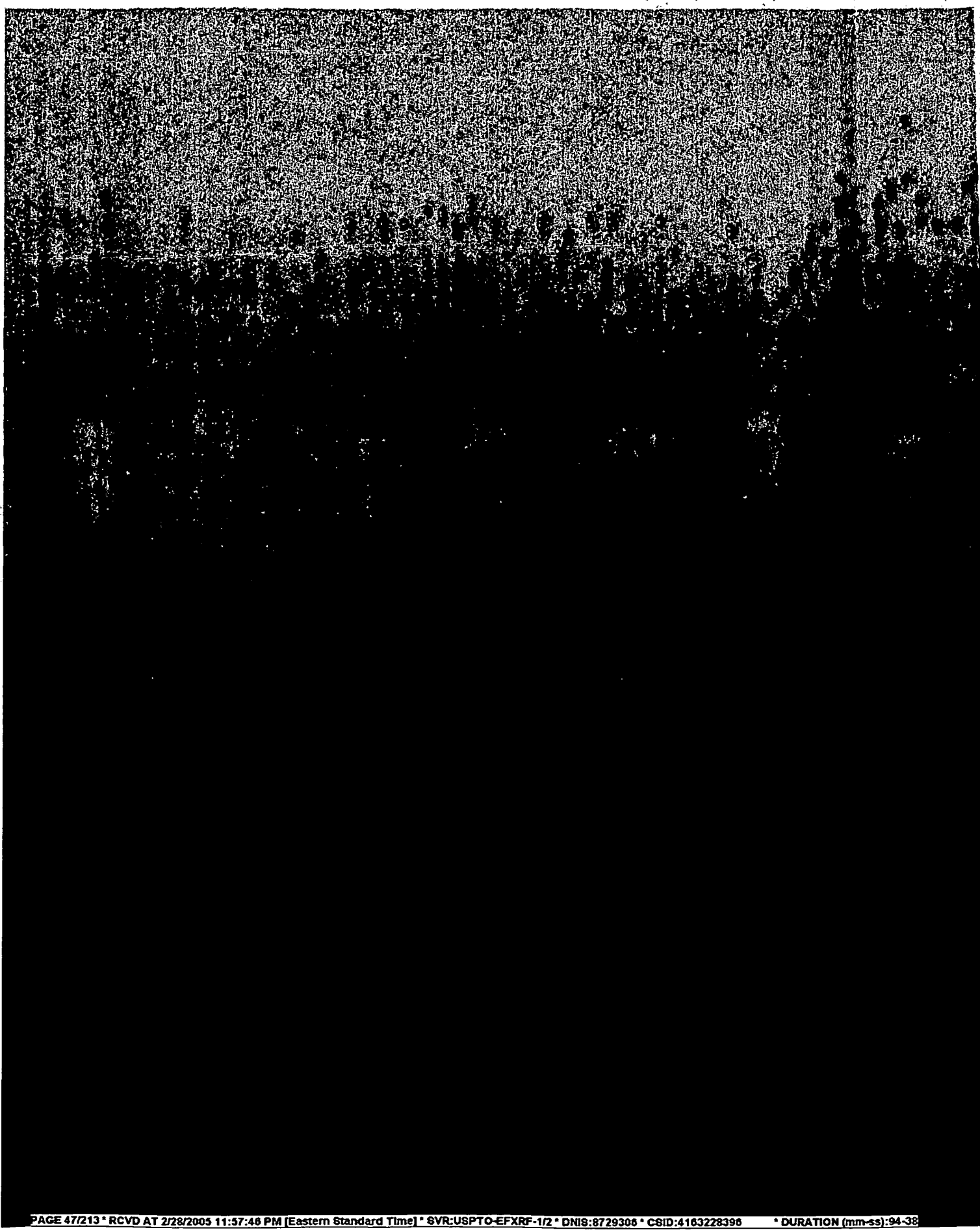
(2) Applicant's invention as defined by claim 115 simultaneously provides the capability to increase the flexibility of meeting deadline constraints when there is flexibility in assigning an offset value for a periodic process.

Neither Dave, nor Dave 2, nor Lindsley show this feature.

(3) Applicant's invention as defined by claim 115 provides the above combination of features, while providing the capability to handle processes that can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are mapped into time slots in the pre-run-time schedule, thus obtaining the advantages of pre-run-time scheduling including ability to use the information in the pre-run-time schedule to achieve greater predictability, ability to handle complex constraints, lower run-time overhead, and in general greatly increase the efficiency of scheduling; and the advantages of run-time scheduling including ability to handle asynchronous processes with very short deadlines that cannot be converted into periodic processes or will waste too much processor capacity if converted into periodic processes, while providing a guarantee that all the constraints, including hard deadlines, will be satisfied before run-time.

Neither Dave, nor Dave 2, nor Lindsley show this combination of features.

A. Applicant's invention achieves **unexpected results**: A system with all the above important features combined together, has never been realized before. The combination



of results achieved by Applicant's invention are new and vastly superior compared to that of Dave, Dave2, and Lindsley or any combination thereof.

B. Applicant's invention is classified in a **crowded art** (a prior art patent cited by the O.A. states that, *"There is a vast amount of literature in the area of scheduling of soft and hard aperiodic tasks", Dave, col. 2, lines 47-48*); therefore, even a small step forward should be regarded as significant.

Applicant therefore submits that claim 115 is patentable under § 102 and § 103 and should be allowed, since they produce new and unexpected results over Dave, Dave2, and Lindsley or any combination thereof.

The Dependent Claims 59-65, 68, 75-76, 78-79, 81, 83-85, 89, 91, 95, 86-88, 90, 92, 94, 97-99, 103-110, 113-114 Are A Fortiori Patentable Over Dave, Dave2, and Lindsley

New dependent claims 117-122, 124-154, 157 incorporate the subject matter of claim 116, claim 115, and 150 and add additional subject matter which makes them a fortiori and independently patentable over these references.

16. The last O.A. rejected dependent claim 60 on Dave, Dave2, and Lindsley. Claim 60 has been cancelled since Applicant considers it to be redundant over the rewritten claims.

17. The last O.A. rejected dependent claim 61 on Dave, Dave2, and Lindsley. Claim 61 has been rewritten as new dependent claim 117 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.

Applicant submits that claim 117 is independently patentable over Dave, Dave2, and Lindsley and any combination thereof. Applicant requests reconsideration of this rejection, as now applicable to claim 117, for the same reasons as given in item 15 above, and in addition, the following reasons:

Claim 117 clearly distinguishes over Dave, Dave2, and Lindsley since it recites: "*executing a set of asynchronous processes that are not mapped to time slots during run-time of the processor at times which do not interfere with execution of processes mapped to time slots in the pre-run-time schedule.*"

Neither Dave, nor Dave2, nor Lindsley do this.

17.1. The last O.A. said that "Lindsley teaches executing a set of non-converted asynchronous processes during run-time of the processor at times which do not interfere with executions of processes contained in the pre-run-time schedule." Applicant submits that the above statement is mistaken because Lindsley does not teach anything related to processes mapped to a pre-run-time schedule. In addition, any possible combination of Dave, Dave2, Lindsley will be inoperative for reasons detailed in item 15.4. above.

17.2. As explained in item 13 above, Dave and Dave2 are co-synthesis algorithms that are performed *off-line* and do not execute tasks on a processor during run-time.

17.3. As explained in item 15 above, Dave and Dave do not show asynchronous processes that are not mapped to time slots in the pre-run-time schedule.

18. The last O.A. rejected dependent claim 62 on Dave, Dave2, and Lindsley. Claim 62 has been rewritten as new dependent claim 118 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant submits that claim 118 is independently patentable over Dave, Dave2, and Lindsley and any combination therefore for the same reasons as given in item 15 above.

Claim 118 clearly distinguishes over Dave, Dave2, and Lindsley since it recites: "*following pre-run-time scheduling and during run-time of the processor, the step of scheduling executions of a set of asynchronous processes that are not mapped to time slots and executions of periodic processes including said new periodic processes that are mapped to time slots such that said predetermined constraints are satisfied.*"

Neither Dave, nor Dave2, nor Lindsley do this.

The last O.A. said that "Dave in view of Lindsley teaches a method as defined in claim 58 including, following pre-run-time scheduling and during run-time of the processor, the step of scheduling executions of a specified set of periodic and asynchronous processes such that said predetermined constraints is satisfied . . . In addition, Dave teaches scheduling tasks to be executed (*"The next step is scheduling which determines the relative orderings of tasks/edges for execution... the algorithm employs a combination of both preemptive and non-preemptive static scheduling" col. 9, lines 65-67 through col. 10, lines 1-5*) Applicant submits that the above statement is mistaken because in the above citation of Dave, as explained in item 3 above, Dave and Dave2 are co-synthesis algorithms that are performed *off-line* and do not execute tasks on a processor during run-time. Thus "scheduling" in Dave and Dave2 refers to offline scheduling of tasks. In addition, any possible combination of Dave, Dave2, and Lindsley will be inoperative for reasons detailed in item 15 above.

19. The last O.A. rejected dependent claim 63 on Dave, Dave2, and Lindsley. Claim 63 has been rewritten as new dependent claim 118 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant submits that claim 118 is independently patentable over Dave, Dave2, and Lindsley and any combination therefore for the same reasons as given in item 15 above, and in addition, the following reasons:

Claim 118 clearly distinguishes over Dave, Dave2, and Lindsley since it recites: *"following pre-run-time scheduling and during run-time of the processor, the step of scheduling executions of a set of asynchronous processes that are not mapped to time slots and executions of periodic processes including said new periodic processes that are mapped to time slots such that said predetermined constraints are satisfied."*

Neither Dave, nor Dave2, nor Lindsley do this.

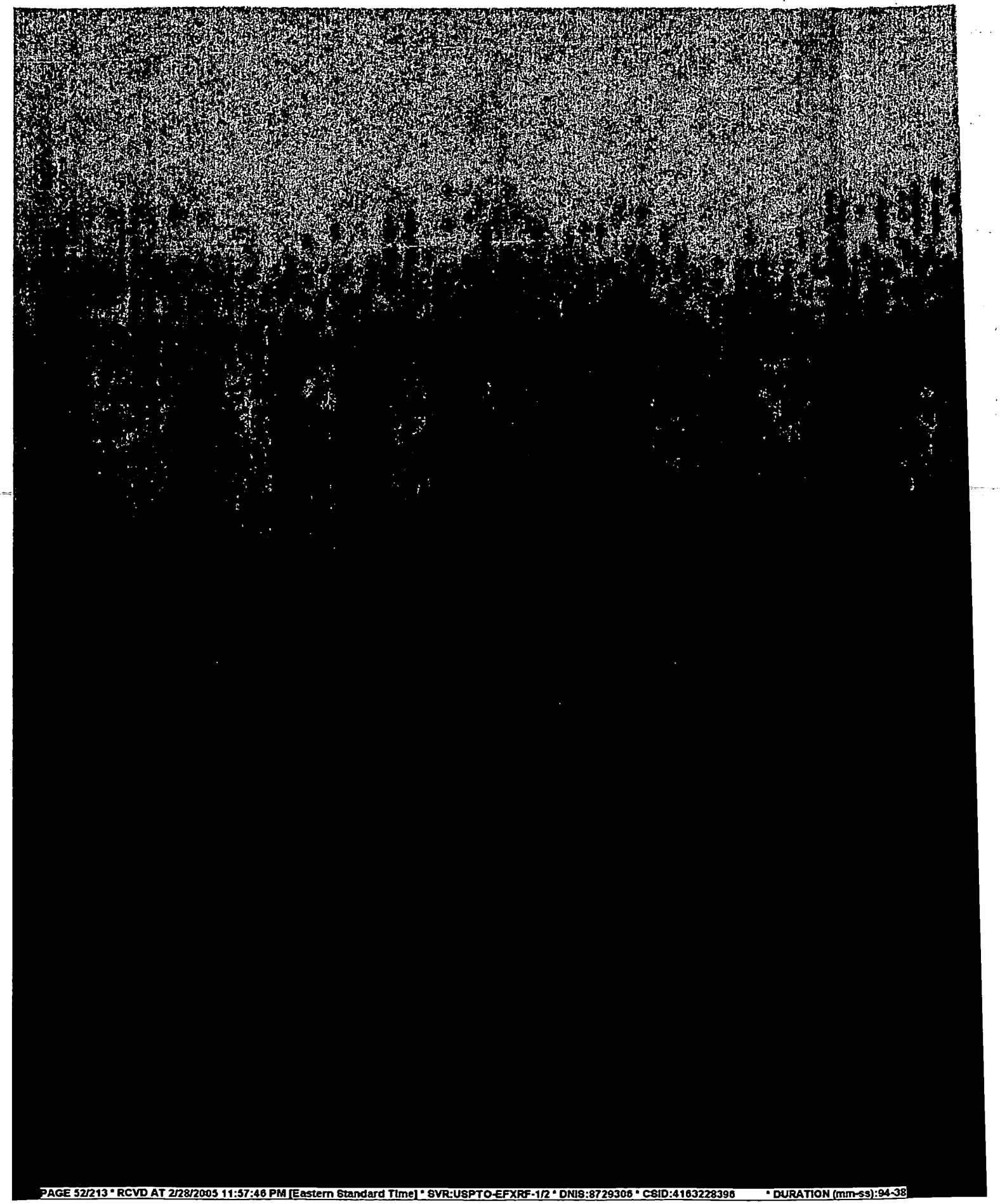
As explained in item 13 and item 15 neither Dave, nor Dave2, nor Lindsley satisfy all the predetermined constraints in claim 118.

(a) Neither Dave, nor Dave schedule asynchronous processes that are not mapped to time slots during run-time.

(b) Lindsley does not have any notion of periodic processes, thus does not schedule new periodic processes that are mapped to time slots.

(c) Lindsley does not have any notion of constraints such as worst-case computation times, deadline, period, etc. Hence Lindsley does not teach satisfying the predetermined constraints.

The last O.A. said that "Dave in view of Lindsley teaches a method as defined in claim 58 including, following pre-run-time scheduling and during run-time of the processor, the step of scheduling executions of a specified set of periodic and asynchronous processes, worst-case computation time, deadline, minimum time between two consecutive requests, and beginning time and end time of every time slot reserved for every periodic process execution in the pre-run-time schedule (see rejection of claim 59-62). Dave in view of Lindsley fails to explicitly teach the specified constraints and relations will be satisfied. However, "Official Notice" is taken that both the concepts and advantages of providing that satisfying all constraints/relations is well known and expected in the art. It would have been obvious to one of ordinary skill in the art at the time the invention was made to include satisfying all constraints/relations to the existing method for the reason of improving accuracy by making sure that no constraints/relations are missed."



Applicant respectfully disagrees with the above "Official Notice", and requests that documentary proof be provided, and the data be stated as specifically as possible, and the facts be supported, under M.P.E.P Section 2144.03 and 37 CFR 1.104(d)(2) for the "Office Notice" position that it would have been obvious to one of ordinary skill in the art at the time the invention was made to include satisfying all constraints/relations in the context of claim 118, that is:

"*scheduling on one or more processors, executions of a plurality of periodic and asynchronous processes, comprising:*

(A)

automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(1) worst-case computation times for periodic processes and asynchronous processes,

(2) period for periodic processes,

(3) minimum time between two consecutive requests for asynchronous processes,

(4) deadline for periodic processes and asynchronous processes,

(5) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(6) precedence relations for periodic processes wherein each precedence relation being defined between a pair of processes comprising a first process and a second process, both said first process and said second process being periodic processes,

said first process precedes said second process, execution of said second process only allowed to start after said first process has completed its execution,

(7) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots, including the step of converting one or more asynchronous processes into corresponding new periodic processes prior to the mapping step, and mapping new periodic processes to time slots in a manner similar to mapping of other periodic processes, such that said predetermined constraints will be satisfied

(B)

during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such that said predetermined constraints will be satisfied"

and

"following pre-run-time scheduling and during run-time of the processor, the step of scheduling executions of a set of asynchronous processes that are not mapped to time slots and executions of periodic processes including said new periodic processes that are mapped to time slots such that said predetermined constraints are satisfied."

for the reason of improving accuracy by making sure that no constraints/relations are missed."

The reasons that Applicant disagrees with the above "Official Notice", include the following:

19.1. Applicant submits that satisfying the particular combination of predetermined constraints (1)-(7) in claim 115 and 118, and further satisfying this particular combination of predetermined constraints under the particular circumstances as defined in claim 115 and 118, that is, during scheduling before run-time, and during run-time, is non-obvious, and would NOT have been obvious to one of ordinary skill in the art at the time the invention was made.

19.2. Applicants submits that the failure of all the references cited by the last O.A., or any combination therefore, to satisfy the particular combination of predetermined constraints (1)-(7) in claim 115 and 118, and further satisfying this particular combination of predetermined constraints under the particular circumstances as defined in claim 115 and 118, that is, during scheduling before run-time, and during run-time, provides further strong evidence that satisfying the predetermined constraints as defined in claim 118, is non-obvious, and would NOT have been obvious to one of ordinary skill in the art at the time the invention was made.

19.2.1. Applicant has practiced the art of scheduling processes with hard-timing constraints for 20 years as a researcher and academic, and in Applicant's experience, being able to provide proof and guarantee before run-time that a scheduling method will satisfy all the specified constraints is the single most critical, most difficult, and most valuable feature to achieve in a safety-critical system with hard timing constraints.

19.2.2. In Applicant's experience, among the vast number of scheduling methods that have been published or patented, scheduling methods that can truly provide proof and guarantee before run-time that a scheduling method will satisfy all the specified constraints are very rare, and correspondingly very few inventors are able to make such a claim. This is especially true when the specified constraints include complex constraints such as exclusion relations, permitted range of offset, precedence, as defined in Applicant's invention.

19.2.3. In Applicant's experience, the capability to provide proof and guarantee before run-time that all the specified constraints will be satisfied should be one of the most important criteria in judging the true usefulness of a real-time scheduling method.

20. The last O.A. rejected dependent claim 64 on Dave, Dave2, and Lindsley. Claim 64 has been rewritten as new dependent claim 119 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant submits that claim 119 is independently patentable over Dave, Dave2, and Lindsley and any combination therefore for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons:

Claim 119 clearly distinguishes over Dave, Dave2, and Lindsley since it recites: "*scheduling, between the beginning time and end time of each of the time slots in the pre-run-time schedule reserved for execution of a corresponding periodic process, time capacity sufficient to complete execution of said corresponding periodic process and additional time capacity sufficient to complete execution of asynchronous processes that are not converted to new periodic processes and hence not mapped to time slots in the pre-run-time schedule in a manner similar to mapping of other periodic processes and have less latitude than said corresponding periodic process in meeting their respective deadlines.*"

Neither Dave, nor Dave2, nor Lindsley do this.

The last O.A. said that "Dave in view of Lindsley teaches scheduling, within the pre-run-time schedule, a difference between the end time and the beginning time of each of said periodic time slots with sufficient time capacity for execution of asynchronous processes that have less latitude than considered ones of periodic processes in meeting their respective deadlines. In addition, Dave teaches that processes in the schedule fit into time slots (*"For each aperiodic task, as explained before, the algorithm positions the execution slots throughout the hyperperiod after scheduling the first execution slot. If the execution slot cannot be allocated at the required instant, the algorithm schedules its at the earliest possible time and repositions the remaining slots to ensure that the deadlines are always met." Dave, col. 12, lines 20-26).* Applicant submits that this is a

misunderstood reference, as the reference clearly does not teach what the O.A. relies upon it as supposedly teaching.

As already explained in item 11.2.1. and 11.2.2, Dave does not teach: "*scheduling, between the beginning time and end time of each of the time slots in the pre-run-time schedule reserved for execution of a corresponding periodic process, time capacity sufficient to complete execution of said corresponding periodic process and additional time capacity sufficient to complete execution of asynchronous processes that are not converted to new periodic processes and hence not mapped to time slots in the pre-run-time schedule in a manner similar to mapping of other periodic processes and have less latitude than said corresponding periodic process in meeting their respective deadlines.*"

21. The last O.A. rejected dependent claim 65 on Dave, Dave2, and Lindsley. Claim 65 has been rewritten as new dependent claim 120 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant submits that claim 120 is independently patentable over Dave, Dave2, and Lindsley and any combination thereof.
Applicant requests reconsideration of this rejection, as now applicable to claim 120, for the same reasons as given in items 15 and 20 above, and in addition, for the following reasons:

Claim 120 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:
"*A method as defined in claim 116, including scheduling, between the beginning time and end time of each of the time slots in the pre-run-time schedule reserved for execution of a corresponding periodic process, time capacity sufficient to complete execution of said corresponding periodic process and additional time capacity sufficient to complete execution of asynchronous processes that are not converted to new periodic processes and hence not mapped to time slots in the pre-run-time schedule in a manner similar to mapping of other periodic processes and have less latitude than said corresponding periodic process in meeting their respective deadlines.*"

Neither Dave, nor Dave2, nor Lindsley do this, for the same reasons as given in item 20 in this amendment.

22. The last O.A. rejected dependent claim 75 on Dave, Dave2, and Lindsley. Claim 75 has been rewritten as new independent claim 124 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant submits that claim 124 is independently patentable over Dave, Dave2, and Lindsley and any combination therefore for the same reasons as given in items 15 and 20 above, and in addition the following reason:

Claim 124 clearly distinguishes over Dave, Dave2, and Lindsley since it recites: "*including, prior to generating the pre-run-time schedule, determining whether each asynchronous process should or should not be converted into a new periodic process, converting a subset of a predetermined set of asynchronous processes having worst-case computation time, deadline, minimum time between two requests constraints, which have been determined to be convertible, into a set of new periodic processes having worst-case computation time, period, deadline, permitted range of offset constraints and reducing possible timing conflicts with other periodic or asynchronous processes with less latitude in meeting their deadlines, by taking into consideration the computation time requirements of the latter processes when determining the deadline of the new periodic process.*"

Neither Dave, nor Dave2, nor Lindsley do this. (This is explained in item 13 of this amendment).

23. The last O.A. rejected dependent claim 76 on Dave, Dave2, and Lindsley. Claim 76 has been rewritten as new dependent claim 125 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant submits that claim 125 is independently patentable over Dave, Dave2, and Lindsley and any combination therefore for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons:

Claim 125 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:
"*prior to generating the pre-run-time schedule, determining whether each asynchronous process should or should not be converted into a new periodic process, converting a subset of a predetermined set of asynchronous processes having a worst-case computation time, deadline, minimum time between two requests constraints which have been determined to be convertible, into a set of new periodic processes having release time, worst-case computation time, period, deadline, permitted range of offset constraints, wherein a permitted range of offset of each new periodic process being a subinterval of an interval or a full interval that begins at the earliest time that the corresponding being converted asynchronous process can make a request for execution, and ends at a time equal to the sum of the earliest time that said being converted asynchronous process can make a request for execution plus the period length of the new periodic process minus one time unit.*"

Neither Dave, nor Dave2, nor Lindsley do this.

The last O.A. said that "In addition, Dave teaches the process being is a subinterval of an interval or a full interval that begins at the earliest time that the corresponding being converted asynchronous process can make a request for execution, and ends at a time equal to the sum of the earliest time *("Each periodic task graph has an earliest start time (est), period, and deadline (do). Each task of a periodic task graph inherits the task graph's period. Each task in a periodic task graph can have a different deadline. Hard*

aperiodic task graphs have a specified deadline which must be met. Aperiodic task graphs are characterized by a parameter, .UPSILON., denoting the minimum time interval between two consecutive instances of an aperiodic task graph. An aperiodic task graph may start at any time." col.5, lines 54-67 through col. 6, lines 1-4). "Official Notice" is taken that both the concept and advantages of providing that making an executing request for the length of a period minus one is well known and expected in the art. It would have been obvious to one of ordinary skill in the art at the time the invention was made to include requesting the length of the period minus one to the existing method in order to ensure that the new process will fit into the time slot for it."

Applicant respectfully disagrees with the above "Official Notice", and requests that documentary proof be provided, and the data be stated as specifically as possible, and the facts be supported, under M.P.E.P Section 2144.03 and 37 CFR 1.104(d)(2) for the "Office Notice" position that both the concept and advantages of providing that making an executing request for the length of a period minus one in the context of claim 125, that is:

"*prior to generating the pre-run-time schedule, determining whether each asynchronous process should or should not be converted into a new periodic process, converting a subset of a predetermined set of asynchronous processes having a worst-case computation time, deadline, minimum time between two requests constraints which have been determined to be convertible, into a set of new periodic processes having release time, worst-case computation time, period, deadline, permitted range of offset constraints, wherein a permitted range of offset of each new periodic process being a subinterval of an interval or a full interval that begins at the earliest time that the corresponding being converted asynchronous process can make a request for execution, and ends at a time equal to the sum of the earliest time that said being converted asynchronous process can make a request for execution plus the period length of the new periodic process minus one time unit.*"

is well known and expected in the art. It would have been obvious to one of ordinary skill in the art at the time the invention was made to include requesting the length of the

period minus one to the existing method in order to ensure that the new process will fit into the time slot for it."

The reasons that Applicant disagrees with the above "Official Notice", are:

(a) Applicant submits that this is a misunderstood reference, as the reference clearly does not teach what the O.A. relies upon it as supposedly teaching, that is, it does not teach "*prior to generating the pre-run-time schedule, determining whether each asynchronous process should or should not be converted into a new periodic process, converting a subset of a predetermined set of asynchronous processes having a worst-case computation time, deadline, minimum time between two requests constraints which have been determined to be convertible, into a set of new periodic processes having release time, worst-case computation time, period, deadline, permitted range of offset constraints, wherein a permitted range of offset of each new periodic process being a subinterval of an interval or a full interval that begins at the earliest time that the corresponding being converted asynchronous process can make a request for execution, and ends at a time equal to the sum of the earliest time that said being converted asynchronous process can make a request for execution plus the period length of the new periodic process minus one time unit.*"

(b) Neither Dave, nor Dave2, nor Lindsley determine whether each asynchronous process should or should not be converted into a new periodic process.

(c). Only a small part of claim 76 is quoted above. The full claim should be, "A method as defined in claim 58 including, prior to generating the pre-run-time schedule, determining whether each asynchronous process should or should not be converted into a new periodic process, converting a subset of a predetermined set of asynchronous processes having a worst-case computation time, minimum time between two requests characteristics and deadline constraints which have been determined to be convertible, into a set of new periodic processes having release time, worst-case computation time, period, deadline, and permitted range of offset constraints, wherein a permitted range of offset of each new periodic process being is a subinterval of an interval or a full interval that begins at the earliest time that the corresponding being converted asynchronous process can make a request for execution, and ends at a time equal to the sum of the

earliest time that said being converted asynchronous process can make a request for execution plus the period length of the new periodic process minus one time unit." Neither Dave, nor Dave2, nor Lindsley teach "a permitted range of offset of each new periodic process being a subinterval of an interval or a full interval that begins at the earliest time that the corresponding being converted asynchronous process can make a request for execution, and ends at a time equal to the sum of the earliest time that said being converted asynchronous process can make a request for execution plus the period length of the new periodic process minus one time unit."

(d) The art of how to convert an asynchronous process to a new periodic process is not merely "synchronization" as suggested by the last O.A., it is one of the most important, yet least understood, and under-studied techniques in the field of real-time computing. Applicant's papers related to real-time computing have been reprinted in two IEEE Computer Society Tutorial collections and are also widely referenced in textbooks on real-time systems. Applicant is internationally well-known as an expert in real-time computing, and Applicant has taken a special interest in this particular technique for over 20 years, yet it had taken Applicant many years before Applicant realized and invented the technique shown by claim 125, hence Applicant can attest to the fact that the technique shown in claim 125 is far from obvious. Applicant is not surprised at all by the fact that no prior art has been found that meets the features shown in claim 125, because, to Applicant's knowledge, up to even today, no one has published a similar invention.

24. The last O.A. rejected dependent claim 78 on Dave, Dave2, and Lindsley. Claim 78 has been rewritten as new dependent claim 127 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.

Applicant submits that claim 127 is independently patentable over Dave, Dave2, and Lindsley and any combination therefore for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons:

Claim 127 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:

"generating the pre-run-time schedule as a feasible two-part pre-run-time-schedule for execution of periodic processes that may have non-zero offsets (a) an initial part which may be of zero length, and (b) a repeating part having length which is equal to a least common multiple of lengths of the periods of the periodic processes, all executions of all periodic processes within a time interval of length equal to the length of the least common multiple of the periodic process periods being included in the repeating part of the pre-run-time schedule, wherein all said predetermined constraints being satisfied for all executions of all periodic processes within said initial part and said repeating part."

Neither Dave, nor Dave2, nor Lindsley do this.

The last O.A. item 24 said that "Referring to claim 78, Dave teaches a method as defined in claim 58, including generating the pre-run-time schedule as a feasible two-part pre-run-time-schedule for execution of periodic processes that may have non-zero offsets (a) an initial part which may be of zero length, and (b) a repeating part having length which is equal to a least common multiple of lengths of the periods of the periodic processes, all executions of all periodic processes within a time interval of length equal to the length of the least common multiple of the periodic process periods being included in the repeating part of the pre-run-time schedule, wherein all said specified constraints being satisfied for all executions of all periodic processes within both said initial part and said repeating part, and using any offset value in a permitted range of offsets of each periodic process, including any offset value in the permitted range of offsets of any new periodic process that may have been converted from an asynchronous process, to generate said feasible pre-run-time schedule. *("The hyperperiod of the system is computed as the least common multiple (LCM) of the periods of the various periodic task graphs in the specification. According to traditional real-time computing theory, a set of periodic tasks graphs has a feasible schedule if and only if it is schedulable in the hyperperiod." col. 7, lines 33-53).* It is notoriously well known in the art of computer programming that variables can be (and most commonly are) set to zero as initial conditions and can change to non-zero values. It would have been obvious to one of ordinary skill in the art at the time the

invention was made to include the feature of initializing variables because it is standard convention/practice in the art of computer science programming. At the initial or starting state, it is common that zero items have occurred and once past the initial state, it is common that non-zero data has occurred, for example."

Applicant respectfully disagrees with the above statements and submits that claim 127 is unobvious and is independently patentable over Dave, Dave2, and Lindsley and any combination therefore for the following reasons:

(a) Neither Dave, nor Dave2, nor Lindsley show a two-part pre-run-time schedule consisting of an initial part and a repeating part in which all the predetermined constraints are satisfied.

(b) Claim 127 does not refer to "the feature of initializing variables", neither does claim 127 rely on or teach "initializing variables". Hence Applicant submits that the last O.A.'s rejection of claim 127 based on a feature that claim 127 does not possess is unjustified.

(c) Prior art that teaches constructing a two-part schedule in the manner suggested by the last O.A. item 24, that is, by simply adding an initial part that does nothing else but performs functions that are unrelated to the limitations of claim 127 such as "initializing variables" to a repeating part, will NOT teach the limitations of claim 127, because a two-part schedule constructed in this manner, will generally fail to satisfy the predetermined constraints when the constraints include permitted set of offset constraints and the periodic processes have non-zero offsets.

(d) Applicant submits that the fact in 12.3. militates in favor of Applicant because it proves that claim 127 produces unexpected results and hence is unobvious.

(e) When periodic processes have non-zero offsets, then often the only correct schedule is a two-part schedule consisting of an initial part and a repeating-part. Dave uses only a "one-part" schedule with length equal to the least common multiple of the process periods. In fact using only a "one-part" schedule with length equal to the least common multiple of the process periods can lead to the

generation of erroneous schedules when periodic processes have non-zero offsets.

The fact that Dave and Dave2 are not aware of this fact alone shows that the concept of a two-part schedule consisting of an initial part and a repeating-part is unobvious.

(f) The validity of the statement "According to traditional real-time computing theory, a set of periodic task graphs has feasible schedule if and only if it is schedulable in the hyperperiod," depends on what constraints on the tasks exist, and what scheduling method is used. It is well known (since 1981) that if the following conditions hold: (a) all periodic tasks are completely preemptable, (b) no tasks have non-zero offsets, and (c) if the scheduling method is earliest-deadline-first, then this statement is true, but when these conditions do not hold, then this statement can be false.

Again, the fact that Dave and Dave2 do not know the latter fact, also in itself shows that the concept of a two-part schedule consisting of an initial part and a repeating-part is unobvious.

(g) When it is unavoidable that the schedule must be a two-part schedule consisting of an initial part and a repeating-part, then there are other previously unsolved problems, such as how to determine at which point does the repeating part begin, and what will be the length of the repeating part under exactly what conditions? This is one of the previously unanswered problems that Applicant's invention has solved. (In fact, if the constraints are different and the scheduling method is different, then the answers to these questions can be quite different.

Again, Dave and Dave2 would not have known this, which again shows that it is unobvious.

(h) The assumptions made in this invention do not include performing unknown random tasks like "initializing non-zero variables". All that is assumed is that each process has a certain "permitted range of offsets, that is, the time intervals during which that process can start its first period. As soon as the first period of a process is started, that process must start regularly doing a specified amount of work in each period.

25. The last O.A. rejected dependent claim 79 on Dave, Dave2, and Lindsley. Claim 79 has been rewritten as new dependent claims 128 and 129 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant submits that claims 128 and 129 are independently patentable over Dave, Dave2, and Lindsley and any combination therefore for the same reasons as given in item 13, item 15 and item 24 above.

Claim 128 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:
"*using any offset value in a permitted range of offsets of each periodic process, including any offset value in the permitted range of offsets of any new periodic process that may have been converted from an asynchronous process, to generate said feasible pre-run-time schedule.*"
Neither Dave, nor Dave2, nor Lindsley do this.

Claim 129 clearly distinguishes over Dave, Dave2, and Lindsley since it includes all the limitations of claim 128 recited above, and further recites:
"*A method as defined in claim 128, wherein said permitted range of offsets of any new periodic process that may have been converted from an asynchronous process is a subinterval of an interval or a full interval that begins at the earliest time that the corresponding being converted asynchronous process can make a request for execution, and ends at a time equal to the sum of the earliest time that said being converted asynchronous process can make a request for execution plus the period length of the new periodic process minus one time unit.*"
Neither Dave, nor Dave2, nor Lindsley do this.

26. The last O.A. rejected dependent claim 81 on Dave, Dave2, and Lindsley. Claim 81 has been rewritten as new dependent claims 131 and 132 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.

Applicant submits that claims 131 and 132 are independently patentable over Dave, Dave2, and Lindsley and any combination therefore for the same reasons as given in item 13 and item 15 above.

Claim 131 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:

"*including using any offset value in a permitted range of offsets of each periodic process, including any offset value in the permitted range of offsets of any new periodic process that may have been converted from an asynchronous process, to generate said feasible pre-run-time schedule.*"

Neither Dave, nor Dave2, nor Lindsley do this.

Claim 132 clearly distinguishes over Dave, Dave2, and Lindsley since it includes all the limitations of claim 131 recited above, and further recites:

"*A method as defined in claim 131, wherein said permitted range of offsets of any new periodic process that may have been converted from an asynchronous process is a subinterval of an interval or a full interval that begins at the earliest time that the corresponding being converted asynchronous process can make a request for execution, and ends at a time equal to the sum of the earliest time that said being converted asynchronous process can make a request for execution plus the period length of the new periodic process minus one time unit.*"

The last O.A., said in item 26 that, "Referring to claim 81, Dave teaches a method as defined in claim 80, including generating said pre-run-time schedule on a plurality of processors ("co-simulation", "multiple procesors", col. 2, lines 17-42)." The above reference to Dave, col. 2, lines 17-42, does not teach "*including using any offset value in a permitted range of offsets of each periodic process, including any offset value in the permitted range of offsets of any new periodic process that may have been converted from an asynchronous process, to generate said feasible pre-run-time schedule.*"

27. The last O.A. rejected dependent claims 83-85, 89, 91, 95 on Dave, Dave2, and Lindsley. Claims 83-85, 89, 91, 95 has been rewritten as new dependent claims 134-136,

137, 138, 141 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.

Applicant requests reconsideration of this rejection, as now applicable to claims 134-136, 137, 138, 141, for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons:

Claim 134 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:
"(a) during run-time, or during a pre-run-time phase, using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to determine, for any point in time, whether there exists a possibility that immediate execution of a particular asynchronous process may cause the execution of any periodic process with less latitude in meeting a deadline of the latter periodic process as compared with latitude of meeting the deadline of said asynchronous process, to be delayed beyond a predetermined time limit, even if the periodic process is not ready for execution at said any point in time, and
(b) during run-time, delaying execution of said asynchronous process if said possibility is found to exist, even if said possibility is the only reason for delaying the execution of said asynchronous process at said any point in time, and even if the delay will cause the processor to be in an idle state for a time interval of non-zero length beginning from said any point in time.".
Neither Dave, nor Dave2, nor Lindsley do this. As shown in item 13 of this amendment, Dave and Dave2 are co-synthesis algorithms that operate off-line. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

Claim 135 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:
"(a) either during run-time, or during a pre-run-time phase, using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to determine, for any point in time, whether there exists a possibility that immediate execution of a particular asynchronous process may cause the execution of any periodic process with less latitude in meeting a deadline of the latter periodic process as compared with latitude of meeting the deadline of said

asynchronous process, to be delayed beyond the end of the time slot of the periodic process in the pre-run-time schedule, even if the periodic process is not ready for execution at said any point in time, and

(b) during run-time, delaying execution of said asynchronous process if said possibility is found to exist, even if said possibility is the only reason for delaying the execution of said asynchronous process at said any point in time, and even if the delay will cause the processor to be in an idle state for a time interval of non-zero length beginning from said any point in time."

Neither Dave, nor Dave2, nor Lindsley do this. As shown in item 13 of this amendment, Dave and Dave2 are co-synthesis algorithms that operate off-line. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

Claim 136 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:

"(a) either during run-time, or during a pre-run-time phase, using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to determine, for any point in time, whether there exists a possibility that immediate execution of a particular asynchronous process may cause the execution of said asynchronous process, or the execution of some other asynchronous process, to extend beyond the beginning of the time slot of any periodic process that has not yet started in the pre-run-time schedule, even if the periodic process is not ready for execution at said any point in time, and

(b) during run-time, delaying execution of said asynchronous process if said possibility is found to exist, even if said possibility is the only reason for delaying the execution of said asynchronous process at said any point in time, and even if the delay will cause the processor to be in an idle state for a time interval of non-zero length beginning from said any point in time."

Neither Dave, nor Dave2, nor Lindsley do this. As shown in item 13 of this amendment, Dave and Dave2 are co-synthesis algorithms that operate off-line. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

Claim 137 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:

"(a) either during run-time, or during a pre-run-time phase, using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to determine, for any point in time, whether there exists a possibility that immediate execution of a particular asynchronous process may cause the execution of any periodic process to be delayed beyond the end of the time slot of that periodic process in the pre-run-time schedule, even if the periodic process is not ready for execution at said any point in time, and
(b) during run-time, delaying execution of said asynchronous process if said possibility is found to exist, even if said possibility is the only reason for delaying the execution of said asynchronous process at said any point in time, and even if the delay will cause the processor to be in an idle state for a time interval of non-zero length beginning from said any point in time."
Neither Dave, nor Dave2, nor Lindsley do this. As shown in item 13 of this amendment, Dave and Dave2 are co-synthesis algorithms that operate off-line. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

Claim 138 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:
"(a) either during run-time, or during a pre-run-time phase, using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to determine, for any point in time, whether there exists a possibility that immediate execution of a particular first asynchronous process may cause execution of any second asynchronous process to be continuously blocked for a duration of the execution of the first asynchronous process and a duration of the execution of any periodic process, when the second asynchronous process has less latitude in meeting the deadline of the second asynchronous process as compared with the latitude of both the first asynchronous process in meeting the deadline of the first asynchronous process and the latitude of the periodic process in meeting the deadline of the periodic process, even if neither the periodic process nor the second asynchronous process are ready for execution at said any point in time, and
(b) during run-time, delaying execution of the first asynchronous process if said possibility is found to exist, even if said possibility is the only reason for delaying the

execution of the first asynchronous process at said point in time, and even if the delay will cause the processor to be in an idle state for a time interval of non-zero length beginning from said any point in time."

Neither Dave, nor Dave2, nor Lindsley do this. As shown in item 13 of this amendment, Dave and Dave2 are co-synthesis algorithms that operate off-line. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

Claim 141 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:

"*A method as defined in claim 115, including, during a pre-run-time phase, determining by simulation a worst-case response time of an asynchronous process corresponding to a feasible pre-run-time schedule of periodic processes consisting of an initial part of the pre-run-time schedule which may be of zero length and a repeating part of the pre-run-time schedule, wherein for each point in time from zero to the end time of the repeating part of the run-time schedule minus one time unit, simulating execution of the asynchronous process using functions used to determine a run-time schedule and recording response time of the asynchronous process under the assumption that the asynchronous process arrives at a point in time under consideration, all other asynchronous processes that can possibly block the asynchronous process arriving at one time unit prior to the point in time under consideration, and all asynchronous processes that have less latitude than latitude of the asynchronous process arriving at the same said point of time under consideration, scheduling executions of periodic processes to start at the beginning time and to complete execution at the end time of their respective time slots in the pre-run-time schedule, wherein whenever the asynchronous process is delayed because it may block execution of some periodic process having less latitude than the latitude of the asynchronous process, or may block execution of some second asynchronous process for the duration of more than one execution of processes having greater latitude as compared with the latitude of the second asynchronous process, all asynchronous*

processes having less latitude as compared with the latitude of the asynchronous process is delayed in order to delay the asynchronous process for a maximum possible amount of

time, thus simulating all possible worst-case scenarios of executions of the asynchronous process."

Neither Dave, nor Dave2, nor Lindsley do this. As shown in item 13 of this amendment, Dave and Dave2 map all asynchronous processes to time slots. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

The last O.A. item 27 said, "Referring to claim 83-85, 89, 91, 95, it is rejected for the same reasons as stated in the rejections of claims 58-64 and 76-79. In addition, Dave also teaches the asynchronous process is to be delayed according to the assumptions ("delay", col. 2, lines 1-15 and 47-67, and "delay constraint", col. 5, lines 25-46.) Applicant respectfully disagrees with the above statement for the following reasons:

Applicant submits that the above O.A. reference to Dave is a mistaken reference:

(a) Applicant could not locate the word "delay" in col. 2, lines 1-15, and 47-67, but has located it instead in col. 3. line 4. In col. 3, line 4, the word delay refers to another reference that bears no relation to Applicant's claims.

(b) The word "delay constraint", col. 5, lines 25-46, as Dave explains, refers to worst-case execution times ("delay constraint, i.e., the worst-case execution time", col. 5, lines 34-35). The word "delay constraint also bears no relation to Applicant's claims.

(c) Claim 134-136, 137, 138, 141 are methods that relate to performing scheduling process executions during run-time. As explained in item 3 of this amendment, both Dave and Dave2 are co-synthesis algorithms that operate off-line, hence neither Dave nor Dave 2 deal with scheduling processes at run-time, and hence neither Dave nor Dave2 teach what claims 134-136, 137, 138, 141 teach.

In addition, Lindsley does not teach anything related to a pre-run-time schedule, nor does Lindsley teach periodic processes. Thus, neither Dave, nor Dave2, nor Lindsley teach the features of claims 134-136, 137, 138, 141.

28. The last O.A. rejected dependent claim 86 on Dave, Dave2, and Lindsley. Claim Applicant has cancelled claim 86 as Applicant considers it to be redundant over other rewritten claims.

29. The last O.A. rejected dependent claim 87 on Dave, Dave2, and Lindsley. Claim Applicant has cancelled claim 87 as Applicant considers it to be redundant over other rewritten claims.

30. The last O.A. rejected dependent claim 88 on Dave, Dave2, and Lindsley. Claim Applicant has cancelled claim 88 as Applicant considers it to be redundant over other rewritten claims.

31. The last O.A. rejected dependent claim 90 on Dave, Dave2, and Lindsley. Claim Applicant has cancelled claim 90 as Applicant considers it to be redundant over other rewritten claims.

32. The last O.A. rejected dependent claim 92 on Dave, Dave2, and Lindsley. Claim Applicant has cancelled claim 92 as Applicant considers it to be redundant over other rewritten claims.

33. The last O.A. rejected dependent claim 94 on Dave, Dave2, and Lindsley. Claim 94 has been rewritten as new dependent claim 140 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant requests reconsideration of this rejection, as now applicable to claim 140, for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons:

Claim 140 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:
"*determining a worst-case response time of an asynchronous process that has not been converted into a periodic process using a formula comprising:*

the sum of worst-case computation times of asynchronous processes and periodic processes that have less or equal latitude as compared with the latitude of the asynchronous process in meeting their respective deadlines,
plus the maximum time that the asynchronous process may possibly be blocked by some asynchronous or periodic process that has greater latitude as compared with the latitude of the asynchronous process in meeting their respective deadlines, plus the worst-case computation time of the asynchronous process multiplied by the number of periodic processes with which the asynchronous process has an exclusion relation."

Neither Dave, nor Dave2, nor Lindsley does this. Dave and Dave2 map all asynchronous processes to time slots. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

34. The last O.A. rejected dependent claim 97 on Dave, Dave2, and Lindsley. Claim 97 has been rewritten as new dependent claim 143 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant requests reconsideration of this rejection, as now applicable to claim 143, for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons:

Claim 143 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:
"*during a pre-run-time phase, generating tables of safe start time intervals for the executions of asynchronous processes, wherein every periodic process in the pre-run-time schedule is scheduled to be executed strictly within its time slot, wherein for every point in time of the pre-run-time schedule, it is determined whether each asynchronous process should be delayed, under the assumption that the actual start time of execution of every periodic process is equal to the beginning time of its time slot, and the actual end time of execution of every periodic process is equal to the end time of its time slot, wherein for every point in time of the pre-run-time schedule, in the event said*

asynchronous process is to be delayed according to the assumptions, the point in time is set to be unsafe and recorded in a corresponding entry in the table for the point in time and said asynchronous process."

Neither Dave, nor Dave2, nor Lindsley do this. Dave and Dave2 map all asynchronous processes to time slots. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

In item 34 of the last O.A., it was said that, "Referring to claim 97, it is rejected for the same reasons as stated in the rejections of claims 58-64 and 76-79. In addition, "Official Notice" is taken that both the concept and advantages of providing tables is well known and expected in the art. It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of data tables to store data to the existing method for having a data structure that organizes the data for access. Dave also teaches the asynchronous process to be delayed according to the assumptions ("delay", col. 2, lines 1-15 and 47-67, and "delay constraint", col. 5, lines 25-46)."

Applicant respectfully disagrees with the above "Official Notice", and requests that documentary proof be provided, and the data be stated as specifically as possible, and the facts be supported, under M.P.E.P Section 2144.03 and 37 CFR 1.104(d)(2) for the "Office Notice" position that providing data tables in the context of claim 143, that is,

> "A method as defined in claim 115, including, during a pre-run-time phase, generating tables of safe start time intervals for the executions of asynchronous processes, wherein every periodic process in the pre-run-time schedule is scheduled to be executed strictly within its time slot, wherein for every point in time of the pre-run-time schedule, it is determined whether each asynchronous process should be delayed, under the assumption that the actual start time of execution of every periodic process is equal to the beginning time of its time slot, and the actual end time of execution of every periodic process is equal to the end

time of its time slot, wherein for every point in time of the pre-run-time schedule, in the event said asynchronous process is to be delayed according to the assumptions, the point in time is set to be unsafe and recorded in a corresponding entry in the table for the point in time and said asynchronous process."

is well known and expected in the art and it would have been obvious to one of ordinary skill in the art at the time the invention was made to include this feature.

The reasons that Applicant disagrees with the above "Official Notice" are as follows:

(a) Applicant could not locate the word "delay" in col. 2, lines 1-15, and 47-67, but has located it instead in col. 3. line 4. In col. 3, line 4, the word delay refers to another reference that bears no relation to Applicant's claims.

(b) The word "delay constraint", col. 5, lines 25-46, as Dave explains, refers to worst-case execution times ("delay constraint, i.e., the worst-case execution time", col. 5, lines 34-35). The word "delay constraint also bears no relation to Applicant's claims.

(c) Claim 143 is a method that relates to performing scheduling process executions at run-time. As explained in item 3 of this amendment, both Dave and Dave2 are co-synthesis algorithms that notoriously well known methods that only operate off-line, hence neither Dave nor Dave2 teach scheduling process executions at run-time, and hence does not teach what claim 143 teaches.

(d) Providing data tables that provide the exact points where each asynchronous process must be delayed in the context of claim 143 where complex constraints including exclusion relations and deadline constraints between asynchronous processes and periodic processes that have been scheduled into time slots must be satisfied, is certainly not obvious and well known in the art.

35. The last O.A. rejected dependent claim 98 on Dave, Dave2, and Lindsley. Claim 98 has been rewritten as new dependent claim 144 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.

Applicant requests reconsideration of this rejection, as now applicable to claim 144, for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons:

Claim 144 clearly distinguishes over Dave, Dave2, and Lindsley since it recites: "*during a pre-run-time phase, generating tables of safe start time intervals for the executions of asynchronous processes, wherein every periodic process is scheduled to be executed strictly within its time slot in the pre-run-time schedule,*
wherein for selected points in time of the pre-run-time schedule, it is determined whether each asynchronous process should be delayed, under the assumption that the actual start time of execution of every periodic process is equal to the beginning time of its time slot, and the actual end time of execution of every periodic process is equal to the end time of its time slot, wherein for selected points in time of the pre-run-time schedule, in the event said asynchronous process is to be delayed according to the assumptions, that point in time is set to be unsafe and recorded in a corresponding entry in the table for the point in time and said asynchronous process."

Neither Dave, nor Dave2, nor Lindsley do this. Dave and Dave2 map all asynchronous processes to time slots. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

In item 35 of the last O.A., it was said that, "Referring to claim 98, it is rejected for the same reasons as stated in the rejections of claims 58-64. In addition, "Official Notice" is taken that both the concept and advantages of providing tables is well known and expected in the art. It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of data tables to store data to the existing method for having a data structure that organizes the data for access. Dave also teaches the asynchronous process to be delayed according to the assumptions ("delay", col. 2, lines 1-15 and 47-67, and "delay constraint", col. 5, lines 25-46)."

Applicant respectfully disagrees with the above "Official Notice", and requests that documentary proof be provided, and the data be stated as specifically as possible, and the facts be supported, under M.P.E.P Section 2144.03 and 37 CFR 1.104(d)(2) for the "Office Notice" position that providing data tables in the context of claim 144, that is,

"A method as defined in claim 115, including, during a pre-run-time phase, generating tables of safe start time intervals for the executions of asynchronous processes, wherein every periodic process is scheduled to be executed strictly within its time slot in the pre-run-time schedule.

wherein for selected points in time of the pre-run-time schedule, it is determined whether each asynchronous process should be delayed, under the assumption that the actual start time of execution of every periodic process is equal to the beginning time of its time slot, and the actual end time of execution of every periodic process is equal to the end time of its time slot, wherein for selected points in time of the pre-run-time schedule, in the event said asynchronous process is to be delayed according to the assumptions, that point in time is set to be unsafe and recorded in a corresponding entry in the table for the point in time and said asynchronous process."

is well known and expected in the art and it would have been obvious to one of ordinary skill in the art at the time the invention was made to include this feature.

The reasons that Applicant disagrees with the above "Official Notice" are as follows:

(a) Applicant could not locate the word "delay" in col. 2, lines 1-15, and 47-67, but has located it instead in col. 3. line 4. In col. 3, line 4, the word delay refers to another reference that bears no relation to Applicant's claims.

(b) The word "delay constraint", col. 5, lines 25-46, as Dave explains, refers to worst-case execution times ("delay constraint, i.e., the worst-case execution time", col. 5, lines 34-35). The word "delay constraint also bears no relation to Applicant's claims.

(c) Claim 144 is a method that relates to performing scheduling process executions at run-time. As explained in item 3 of this amendment, both Dave and Dave2 are co-synthesis algorithms that notoriously well known methods that only operate

off-line, hence neither Dave nor Dave2 teach scheduling process executions at run-time, and hence does not teach what claim 144 teaches.

(d) Providing data tables that provide the exact points where each asynchronous process must be delayed in the context of claim 144 where complex constraints including exclusion relations and deadline constraints between asynchronous processes and periodic processes that have been scheduled into time slots must be satisfied, is certainly not obvious and well known in the art.

36. The last O.A. rejected dependent claim 99 on Dave, Dave2, and Lindsley. Claim 99 has been rewritten as new dependent claim 145 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant requests reconsideration of this rejection, as now applicable to claim 145, for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons:

Claim 145 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:
"*during run-time, detecting at least one event of, at any point in time, whether some asynchronous process has arrived by said point in time, whether some asynchronous process or periodic process has completed its computation at said point in time, and whether said point in time is both the release time and beginning time of a time slot in the pre-run-time schedule for some periodic process, and activating a run-time scheduler at said point in time, and should said at least one event have occurred, determining whether any asynchronous process that has arrived but has not yet been completed should be delayed or immediately put into execution, and including the further step of delaying execution of a first asynchronous process when execution of some other asynchronous process that excludes a periodic process with less or equal latitude as compared with the latitude of the first asynchronous process in meeting their respective deadlines has already started but has not yet been completed.*"

Neither Dave, nor Dave2, nor Lindsley do this. Dave and Dave2 are co-synthesis algorithms that operate off-line. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

37. The last O.A. rejected dependent claim 103 on Dave, Dave2, and Lindsley. Claim 103 has been rewritten as new dependent claim 149 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant requests reconsideration of this rejection, as now applicable to claim 149, for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons:

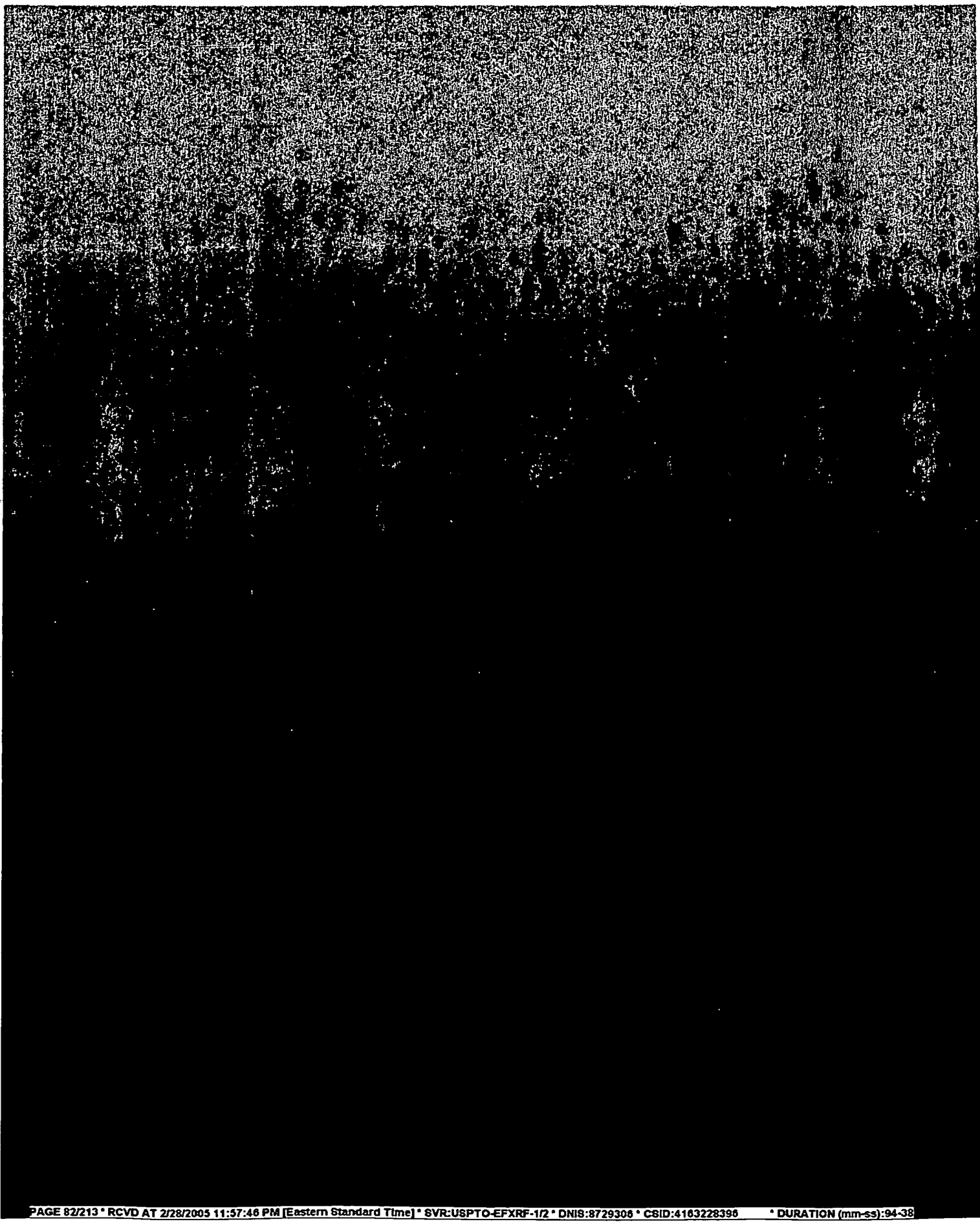
Claim 149 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:
"*during a pre-run-time phase, determining by simulation a worst-case response time of an asynchronous process corresponding to a feasible pre-run-time schedule of periodic processes, wherein for each point in time from zero to the end time of the repeating part of the run-time schedule minus one time unit, simulating execution of said asynchronous process using functions used to determine a run-time schedule and recording response time of said asynchronous process under the assumption that said asynchronous process arrives at a point in time under consideration, all other asynchronous processes that can possibly block said asynchronous process arriving at one time unit prior to the point in time under consideration, and all asynchronous processes that have less latitude than latitude of said asynchronous process arriving at the same said point of time under consideration, scheduling executions of periodic processes to start at the beginning time and to complete execution at the end time of their respective time slots in the pre-run-time schedule, simulating all possible worst-case scenarios of executions of said asynchronous process.*"

Neither Dave, nor Dave2, nor Lindsley do this. Dave and Dave2 map all asynchronous processes to time slots. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

38. The last O.A. rejected dependent claim 104 on Dave, Dave2, and Lindsley. Claim 104 has been rewritten as new dependent claim 115 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant requests reconsideration of this rejection, as now applicable to claim 115, for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons:

Claim 115 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:
"*scheduling on one or more processors, executions of a plurality of periodic and asynchronous processes, comprising:*
(A)
automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,
including satisfaction of predetermined constraints comprising

- *(1) worst-case computation times for periodic processes and asynchronous processes,*
- *(2) period for periodic processes,*
- *(3) minimum time between two consecutive requests for asynchronous processes,*
- *(4) deadline for periodic processes and asynchronous processes,*
- *(5) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower*



bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(6) precedence relations for periodic processes wherein each precedence relation being defined between a pair of processes comprising a first process and a second process, both said first process and said second process being periodic processes, said first process precedes said second process, execution of said second process only allowed to start after said first process has completed its execution,

(7) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots, including the step of converting one or more asynchronous processes into corresponding new periodic processes prior to the mapping step, and mapping new periodic processes to time slots in a manner similar to mapping of other periodic processes, such that said predetermined constraints will be satisfied

(B)

during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such that said predetermined constraints will be satisfied."

Neither Dave, nor Dave2, nor Lindsley do this.

Neither Dave, nor Dave2, nor Lindsley teach satisfying precedence relations between the periodic processes during run-time while satisfying the combination of predetermined constraints (1)-(7) as defined in claim 115.

The last O.A. said that "Dave fails to explicitly teach a method as defined in claim 58, wherein said predetermined constraints and relations further comprise precedence relations. However, "Official Notice" is taken that both the concept and advantages of providing that predetermined constraints can be comprised of precedence relations is well known and expected in the art. It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of having precedence constraints as predetermined constraints to the existing method for the reason of using current/recent data into the system."

Applicant respectfully disagrees with the above "Official Notice", and requests that documentary proof be provided, and the data be stated as specifically as possible, and the facts be supported, under M.P.E.P Section 2144.03 and 37 CFR 1.104(d)(2) for the "Office Notice" position that "both the concept and advantages of providing that predetermined constraints can be comprised of precedence relations is well known and expected in the art. It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of having precedence constraints as predetermined constraints to the existing method for the reason of using current/recent data into the system."

The reasons that Applicant disagrees with the above "Official Notice" are as follows:
1. Applicant's invention is classified in a **crowded art** (a prior art patent cited by the O.A. states that, *"There is a vast amount of literature in the area of scheduling of soft and hard aperiodic tasks", Dave, col. 2, lines 47-48*); therefore, even a small step forward should be regarded as significant.
2. How to integrate precedence constraints with many other types of complex constraints while at the same time providing proof and guarantee that all specified constraints will be met before run-time can be very complex and unobvious. This is underscored by the fact

that, the most widely known and most intensively studied techniques of scheduling, priority scheduling, still has not been able to provide a proof and guarantee that all specified constraints will be met before run-time when the specified constraints include general precedence relations combined with other complex constraints such as exclusion constraints and offset constraints despite more than four decades of intensive research.

39. The last O.A. rejected dependent claim 105, and 107 on Dave, Dave2, and Lindsley. Claim 105 and 107 have been rewritten as new dependent claims 115 and 124 to define patentably over Dave, Dave2, and Lindsley and any combination thereof. Applicant requests reconsideration of this rejection, as now applicable to claim 115 and 124, for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons:

Claim 124 clearly distinguishes over Dave, Dave2, and Lindsley since it recites: "*prior to generating the pre-run-time schedule, determining whether each asynchronous process should or should not be converted into a new periodic process, converting a subset of a predetermined set of asynchronous processes having worst-case computation time, deadline, minimum time between two requests constraints, which have been determined to be convertible, into a set of new periodic processes having worst-case computation time, period, deadline, permitted range of offset constraints and reducing possible timing conflicts with other periodic or asynchronous processes with less latitude in meeting their deadlines, by taking into consideration the computation time requirements of the latter processes when determining the deadline of the new periodic process.*"

Neither Dave, nor Dave2, nor Lindsley do this. Dave and Dave2 map all asynchronous processes to time slots. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

The last O.A. rejected dependent claim 106 on Dave, Dave2, and Lindsley. Claim 106 has been rewritten as new dependent claim 119 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant submits that claim 119 is independently patentable over Dave, Dave2, and Lindsley and any combination therefore for the same reasons as given in item 13 and item 15 above.

Applicant requests reconsideration of this rejection, as now applicable to claim 119, for the following reasons:

Claim 119 clearly distinguishes over Dave, Dave2, and Lindsley since it recites: "*scheduling, between the beginning time and end time of each of the time slots in the pre-run-time schedule reserved for execution of a corresponding periodic process, time capacity sufficient to complete execution of said corresponding periodic process and additional time capacity sufficient to complete execution of asynchronous processes that are not converted to new periodic processes and hence not mapped to time slots in the pre-run-time schedule in a manner similar to mapping of other periodic processes and have less latitude than said corresponding periodic process in meeting their respective deadlines.*"

Neither Dave, nor Dave2, nor Lindsley do this.
Lindsley does not teach any concept of periodic processes. Neither Dave nor Dave2 teach the above feature of claim 119 as explained in item 39.

The last O.A. said that "Dave in view of Lindsley teaches scheduling, within the pre-run-time schedule, a difference between the end time and the beginning time of each of said periodic time slots with sufficient time capacity for execution of asynchronous processes that have less latitude than considered ones of periodic processes in meeting their respective deadlines. In addition, Dave teaches that processes in the schedule fit into time slots (*"For each aperiodic task, as explained before, the algorithm positions the execution slots throughout the hyperperiod after scheduling the first execution slot. If the*

execution slot cannot be allocated at the required instant, the algorithm schedules its at the earliest possible time and repositions the remaining slots to ensure that the deadlines are always met." Dave, col. 12, lines 20-26). Applicant submits that this is a misunderstood reference, as the reference clearly does not teach what the O.A. relies upon it as supposedly teaching, that is, it does not teach the feature in claim 119. Applicant has discussed this in more detail in item 11.2.1. and 11.2.2. above

40. The last O.A. rejected dependent claim 109 on Dave, Dave2, and Lindsley. Claim 109 have been rewritten as new dependent claims 125 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant requests reconsideration of this rejection, as now applicable to claim 125, for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons:

Claim 125 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:
"*prior to generating the pre-run-time schedule, determining whether each asynchronous process should or should not be converted into a new periodic process, converting a subset of a predetermined set of asynchronous processes having a worst-case computation time, deadline, minimum time between two requests constraints which have been determined to be convertible, into a set of new periodic processes having release time, worst-case computation time, period, deadline, permitted range of offset constraints, wherein a permitted range of offset of each new periodic process being a subinterval of an interval or a full interval that begins at the earliest time that the corresponding being converted asynchronous process can make a request for execution, and ends at a time equal to the sum of the earliest time that said being converted asynchronous process can make a request for execution plus the period length of the new periodic process minus one time unit.*"

Neither Dave, nor Dave2, nor Lindsley do this. Dave and Dave2 map all asynchronous processes to time slots. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

41. The last O.A. rejected dependent claim 110 on Dave, Dave2, and Lindsley. Claim 110 has been rewritten as new dependent claim 127 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant requests reconsideration of this rejection, as now applicable to claim 127, for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons:

Claim 127 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:
"*including generating the pre-run-time schedule as a feasible two-part pre-run-time-schedule for execution of periodic processes that may have non-zero offsets (a) an initial part which may be of zero length, and (b) a repeating part having length which is equal to a least common multiple of lengths of the periods of the periodic processes, all executions of all periodic processes within a time interval of length equal to the length of the least common multiple of the periodic process periods being included in the repeating part of the pre-run-time schedule, wherein all said predetermined constraints being satisfied for all executions of all periodic processes within said initial part and said repeating part.*"

Neither Dave, nor Dave2, nor Lindsley do this. Dave and Dave2 only use a single repeating schedule. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

42. The last O.A. rejected dependent claim 113 on Dave, Dave2, and Lindsley. Claim 113 has been rewritten as new dependent claims 151 to define patentably over Dave, Dave2, and Lindsley and any combination thereof.
Applicant requests reconsideration of this rejection, as now applicable to claim 151, for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons:

Claim 151 clearly distinguishes over Dave, Dave2, and Lindsley since it recites: "*prior to the mapping step, converting one or more asynchronous processes having a worst-case computation time, deadline, minimum time between two requests constraints, into a set of new periodic processes, and reducing possible timing conflicts with other periodic or asynchronous processes with less latitude in meeting their deadlines, by* taking into consideration the computation time requirements of the latter processes when determining the deadline of each of the new periodic processes, *and mapping the new periodic process in a manner similar to mapping of other periodic processes.*"

Neither Dave, nor Dave2, nor Lindsley do this. Dave and Dave2 do not take into consideration the computation time requirements of processes with less latitude when determining the deadline of each of the new periodic processes. Lindsley has no knowledge of a pre-run-time schedule or quantitative timing constraints.

43. The Rejection Of Claim 77 Under 35 USC § 103 On Dave (US 6,178,542 B1), Lindsley (US 6,430,593 B1), And Matsumoto (US5,448, 732) Is Overcome

44. The last O.A. rejected dependent claim 77 on Dave, Lindsley and Matsumoto. Claim 77 has been rewritten as new dependent claim 126 to define patentably over Dave, Lindsley and Matsumoto and any combination thereof. Applicant requests reconsideration of this rejection, as now applicable to claim 126, for the following reasons:

(1) There is no justification, in Dave, Lindsley, and Matsumoto, or in any other prior art separate from Applicant's disclosure, which suggests that these references be combined, much less combined in the manner proposed.

(2) The proposed combination would not be physically possible or operative.

(3) Even if Dave, Lindsley, and Matsumoto were to be combined in the manner proposed, the proposed combination would not show all of the novel physical features of claim 126.

(4) The novel features of claim 126 produce new and unexpected results and hence are unobvious and patentable over these references.

44.1. Dave, Lindsley, and Matsumoto Do Not Contain Any Justification To Support Their Combination, Much Less In The Manner Proposed

With regard to the proposed combination of Dave, Lindsley, and Matsumoto, it is well known that in order for any prior-art references themselves to be validly combined for use in a prior-art § 103 rejection, *the references themselves* (or some other prior art) must suggest that they be combined. E.g., as was stated in In re Sernaker, 217 U.S.P.Q. 1, 6 (C.A.F.C. 1983):

> "[P]rior art references in combination do not make an invention obvious unless something in the prior art references would suggest the advantage to be derived from combining their teachings."

That the suggestion to combine the references should not come from applicant was forcefully stated in Orthopedic Equipment Co. v. United States, 217 U.S.PQ. 193, 199 (CAFC 1983):

> "It is wrong to use the patent in suit [here the patent application] as a guide through the maze of prior art references, combining the right references in the right way to achieve the result of the claims in suit [here the claims pending]. Monday morning quarterbacking is quite proper when resolving the question of nonobviousness in a court of law [here the PTO]."

As was further stated in Uniroyal, Inc. v. Rudkin-Wiley Corp., 5 U.S.P.Q.2d 1434 (C.A.F.C. 1988), "[w]here prior-art references require selective combination by the court to render obvious a subsequent invention, there must be some reason for the combination other than the hindsight gleaned from the invention itself. . . . *Something in the prior art must suggest the desirability and thus the obviousness of making the combination.*" [Emphasis supplied.]

In line with these decisions, the Board stated in Ex Parte Levengood, 28 U.S.P.Q.2d 1300 [P.T.O.B.A.&]. 1993):

"In order to establish a *prima facie* case of obviousness, it is necessary for the examiner to present evidence, preferably in the form of some teaching, suggestion, incentive or inference in the applied prior art, or in the form of generally available knowledge, that one having ordinary skill in the art *would have been led to* combine the relevant teachings of the applied references in the proposed manner to arrive at the claimed invention. . . . That which is within the capabilities of one skilled in the art is not synonymous with obviousness, ... That one can *reconstruct* and/or explain the theoretical mechanism of an invention by means of logic and sound scientific reasoning does not afford the basis for an obviousness conclusion unless that logic and reasoning also supplies sufficient impetus to have led one of ordinary skill in the art to combine the teachings of the references to make the claimed invention. . . . Our reviewing courts have often advised the Patent and Trademark Office that it can satisfy the burden of establishing a *prima facie* case of obviousness only be showing some objective teaching in either the prior art, or knowledge generally available to one of ordinary skill in the art, that 'would lead' that individual 'to combine the relevant teachings of the references." ... Accordingly, an examiner cannot establish obviousness by locating references which describe various aspects of a patent application's invention without also providing evidence of the motivating force which would impel one skilled in the art to do what the patent applicant has done."

In the present case, there is no reason given in the last O.A. to support the proposed combination, other than the statement *"Referring to claim 77, Dave fails to explicitly* teach prior to generating the pre-run-time schedule, determining whether each asynchronous process should or should not be converted into a new periodic process by a ratio or processing capacity. However, Matsumoto teaches a method of determining whether each asynchronous process should or should not be converted into a new periodic process by calculating whether a ratio of processing capacity of the processor

which is required to be reserved for new periodic processes, to processor capacity that is required for the asynchronous process if left unconverted, exceeds a predetermined threshold value. *("Each of [1], [2], and [3] is a condition for improving the theoretical effectiveness, and each of [4] and [5] is a condition for doing the same by determining "n" heuristically, or from experience. Depending on the application which is running, "n" is adjusted in order to improve efficiency. With respect to conditions [4] and [5], instead of the number of processes waiting for synchronization, the ratio of the number of processors in the group to the number of processors waiting for synchronization in the group is used," col. 6, lines 25-35*). It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of synchronizing with respects to a ratio value of processing capacity for the reason of increasing the control of the system. This ratio tells the processor when it can stop waiting for synchronization to begin, for example. As mentioned earlier, it is common knowledge in the art of task management and process synchronization that converting asynchronous processes to synchronous ones is merely synchronization. In addition, Dave in view of Lindsley, and in further view of Matsumoto fail to explicitly teach using predetermined thresholds to determine a state in change. However, "Official Notice" is taken that both the concept and advantages of providing that the use of thresholds is well known and expected in the art. It would have been obvious to one of ordinary skill in the art at the time the invention was made to include thresholds to the existing method for the reason of increasing the control by being able to set limits or boundaries which determine one state over another. In this specific case, synchronization would begin after the threshold is reached." (Last O.A.)

Applicant respectfully disagrees with the above "Official Notice", and requests that documentary proof be provided, and the data be stated as specifically as possible, and the facts be supported, under M.P.E.P Section 2144.03 and 37 CFR 1.104(d)(2) for the "Office Notice" position that "the use of thresholds is well known and expected in the art. It would have been obvious to one of ordinary skill in the art at the time the invention was made to include thresholds to the existing method for the reason of increasing the

control by being able to set limits or boundaries which determine one state over another. In this specific case, synchronization would begin after the threshold is reached."

1. Matsumoto fails to show any of the important features of claim 126. The quoted phrases, "*of processes waiting for synchronization, the ratio of the number of processors in the group to the number of processors waiting for synchronization in the group is used,"(* Matsumoto *col. 6, lines 25-35*) bear no relation to converting an asynchronous process to a new periodic process. It does not show how to calculate the processor capacity that is required for the asynchronous process if left unconverted, or the processor capacity which is required to be reserved for the new periodic process. Matsumoto fails to show any of the features of claims 115 and 124, on which 160 is based. Matsumoto does not even have the notion of timing constraints, such as periods, worst-case computation times, deadlines, etc.

2. The art of how to convert an asynchronous process to a new periodic process is not merely "synchronization" as suggested by the last O.A., it is one of the most important, yet least understood, and under-studied techniques in the field of real-time computing. Applicant's papers related to real-time computing have been reprinted in two IEEE Computer Society Tutorial collections and are also widely referenced in textbooks on real-time systems. Applicant is internationally well-known as an expert in real-time computing, and Applicant has taken a special interest in this particular technique for over 20 years, yet it had taken Applicant many, many years before Applicant realized and invented the technique shown by claim 126, hence Applicant can attest to the fact that the technique shown in claim 126 is far, far from obvious.

3. As can been seen in Fig. 26 of the drawings, and paragraph [0174], determining the ratio of processing capacity of the processor which is required to be reserved for new periodic processes, to processor capacity that is required for the asynchronous process if left unconverted, requires an elaborate procedure that is far from obvious. Hence Applicant is not surprised at all by the fact that no prior art has been found that meets the features shown in claim 126, because, to Applicant's knowledge, up to even today, no one has published a similar invention.

However, the fact that the words "processors waiting for synchronization", "the ratio of the number of processors" "asynchronous" appear in the reference is not sufficient to gratuitously and selectively combine parts of one reference (Matsumoto's multiprocessor system) with another two references in order to meet Applicant's novel claimed combination of features.

Note that:

44.1.1. The meaning of the term "synchronous" in Lindsley is inherently different from the meaning of "periodic" in Applicant's invention as defined by claim 126 as shown below:

44.1.1.(a) Lindsley uses the term "synchronous" when describing "synchronous task commands". *("The TSA accepts commands from tasks called 'synchronous' task commands. These commands are synchronous from the point of view that the task may not issue another synchronous task command until the previous synchronous task command has been completed. It is allowable for the task to perform other activity after issuing a synchronous task command as long as the task verifies that the previous task has been completed prior to issuing another synchronous task command." Lindsley, col. 6, lines 47-55.)* It is clear that in Lindsley there is no periodic constraint on "synchronous" task commands, i.e., synchronous task commands are not constrained to execute strictly once in each fixed period of time.

44.1.1.(b) In contrast, in Applicant's invention as defined by claim 126, "a periodic process consists of a computation that is executed repeatedly, once in each fixed period of time. (Applicant's specification, paragraph [0113]). Applicant's definition of a periodic process, is adopted universally in the field of real-time computing and is clearly different from the meaning of "synchronous" as defined in Lindsley.

44.1.2. The meaning of the term "*synchronization*" in Lindslev is also inherently different from the meaning of "*converting an asynchronous process to a new periodic process*" in Applicant's invention as defined by claim 126 as shown below:

44.1.2.(a) In Lindsley, tasks are synchronized using "events" which are certainly not always periodic.*("Tasks are typically synchronized using "events". An event is used to indicate that an activity has taken place, such as data arrival, time-out, etc. Thus, an event may indicate execution of a task, an interrupt service routine or the like. Events are counted using semaphores. Semaphores synchronize the event producer and the event consumer ... " Lindsley, col. 2, lines 4-10.)*
("If a task that processes data buffers pends for the semaphore that represents data buffers, the task is synchronized to the data buffer generation. If data buffers are available, the semaphore count is greater than zero. Task pend requests on the semaphore a llow the task to continue. If data buffers are not available, the semaphore count is less than or equal to zero, the task does not have data for processing and will block execution". Lindsley, col. 2, lines 52-61.)
44.1.2.(b) In contrast, in Applicant's invention as defined by claim 126, "*converting an asynchronous process to a new periodic process*" means converting a process that can be executed at random times, to a process that will be executed once in each fixed period of time.

44.1.3.(a) In Matsumoto, as with Lindsley, there is no notion of periodic processes, no notion of deadlines or any kind of timing constraints.
The above evidence, clearly shows that the last O.A.'s reason for combining Dave, Lindsley, and Matsumoto to find obviousness of claim 126, i.e., *("It is common knowledge in the art of task management and process synchronization that converting an asynchronous process to a new periodic process (or a synchronous process) is known as synchronization,"* is totally unfounded.

Applicant therefore submits that combining Dave, Lindsley, and Matsumoto is not legally justified and is therefore improper. Thus Applicant submits that the rejection on these references is also improper and should be withdrawn.

Applicant respectfully requests, if the claims are again rejected upon any combination of references, that the Examiner include an explanation, in accordance with M.P.E.P. § 706.02, Ex parte Clapp, 27 U.S.P.Q. 972 (P.O.B.A. 1985), and Ex parte Levengood, supra, a "factual basis to support his conclusion that it would have been obvious" to make the combination,

44.2. Even if Dave, Lindsley, and Matsumoto Were To Be Combined In The Manner Proposed, The Proposed Combination Would Not Show All The Novel Features Of Claim 126.

However even if the combination of Dave, Lindsley, and Matsumoto were justified, claim 126 would still have novel and unobvious features over the proposed combination.

44.2.1. Neither Dave, nor Lindsley, nor Matsumoto, nor any possible combination thereof show the feature of automatically generating a pre-run-time schedule in which predetermined constraints comprising worst-case computation time, period, minimum time between two consecutive exclusion relations, deadline, permitted range of offset constraints, precedence relations are satisfied. (Discussion on Dave's lack of ability to satisfy exclusion constraints is provided in item 11.1. above and is equally applicable to Dave and Dave2 and is hereby included here by reference.)

Claim 126 clearly distinguishes over Dave, Lindsley, Matsumoto or any possible combination thereof, since it recites

"automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time

axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(1) worst-case computation times for periodic processes and asynchronous processes,

(2) period for periodic processes,

(3) minimum time between two consecutive requests for asynchronous processes,

(4) deadline for periodic processes and asynchronous processes,

(5) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(6) precedence relations for periodic processes wherein each precedence relation being defined between a pair of processes comprising a first process and a second process, both said first process and said second process being periodic processes, said first process precedes said second process, execution of said second process only allowed to start after said first process has completed its execution,

(7) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process

including the step of converting one or more asynchronous processes into corresponding new periodic processes prior to the mapping step, and mapping new periodic processes to time slots in a manner similar to mapping of other periodic processes, such that said predetermined constraints will be satisfied".

Neither Dave, nor Lindsley, nor Matsumoto nor any possible combination thereof, show the feature of *automatically generating a pre-run-time schedule including satisfaction of predetermined constraints comprising "exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation".*
(Discussion on Dave's lack of this feature is provided in item 11.1. above and is equally applicable to Dave and Dave2, and is hereby included here by reference.)

Note that Lindsley performs synchronization only at run-time, NOT before run-time, so it is not capable of satisfying any constraints before run-time, which Applicant's invention, as defined by claim 126 would be capable of doing.

44.2.2. Neither Dave, nor Lindsley, nor Matsumoto, nor any possible combination thereof show the feature of permitted range of offset constraints for periodic processes

Claim 126 clearly distinguishes over any combination of Dave, Lindsley, Matsumoto since it recites

"including satisfaction of predetermined constraints comprising

...

(4) *permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value, n,"*

Neither Dave, nor Lindsley, nor Matsumoto, nor any possible combination thereof show the feature of satisfying predetermined constraints comprising "*permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value, n,*"

44.2.3. Neither Dave, nor Lindsley, nor Matsumoto, nor any possible combination thereof show the feature of determining whether each asynchronous process should or should not be converted into a new periodic process by calculating whether a ratio of processing capacity of the processor which is required to be reserved for new periodic processes, to processor capacity that is required for the asynchronous process if left unconverted, exceeds a predetermined threshold value.

Claim 124 clearly distinguishes over any combination of Dave, Lindsley, Matsumoto since it recites

"A method as defined in claim 115 including, prior to generating the pre-run-time schedule, determining whether each asynchronous process should or should not be converted into a new periodic process, converting a subset of a predetermined set of asynchronous processes having worst-case computation time, minimum

time between two requests characteristics and deadline constraints, which have been determined to be convertible, into a set of new periodic processes having worst-case computation time, period, deadline, and permitted range of offset constraints, and reducing possible timing conflicts with other periodic or asynchronous processes with less latitude in meeting their deadlines, by taking into consideration the computation time requirements of the latter processes when determining the deadline of the new periodic process".

Claim 126 recites

A method as defined in claim 124, in which the determining step is performed by calculating whether a ratio of processing capacity of the processor which is required to be reserved for new periodic processes, to processor capacity that is required for the asynchronous process if left unconverted, exceeds a predetermined threshold value.

Neither Dave, nor Lindsley, nor Matsumoto, nor any possible combination thereof show the feature of satisfying predetermined constraints comprising "*determining whether each asynchronous process should or should not be converted into a new periodic process by calculating whether a ratio of processing capacity of the processor which is required to be reserved for new periodic processes, to processor capacity that is required for the asynchronous process if left unconverted, exceeds a predetermined threshold value.*

44.2.4. Matsumoto Fails to Show Any of the Important Features of Claim 126

Claim 126 clearly distinguishes over Dave, Lindsley and Matsumoto since claim 126 includes all the limitations of claim 124, and claim 124 recites:

"*prior to generating the pre-run-time schedule, determining whether each asynchronous process should or should not be converted into a new periodic process, converting a subset of a predetermined set of asynchronous processes having worst-case computation time, deadline, minimum time between two requests constraints, which have been*

possible timing conflicts with other periods of asynchronous processes with less latitude in meeting their deadlines, by taking into consideration the computation time requirements of the latter processes when determining the deadline of the new periodic process."

and claim 126 further recites:

"*the determining step comprises*

(1) calculating a first processor capacity that is required for the asynchronous process if left unconverted,

(2) calculating a second processor capacity which is required to be reserved for the new periodic process,

(3) determining whether the ratio of said first processor capacity to said second processor capacity exceeds a predetermined threshold value."

The last O.A. said that, "*Referring to claim 77, Dave fails to explicitly* teach prior to generating the pre-run-time schedule, determining whether each asynchronous process should or should not be converted into a new periodic process by a ratio or processing capacity. However, Matsumoto teaches a method of determining whether each asynchronous process should or should not be converted into a new periodic process by calculating whether a ratio of processing capacity of the processor which is required to be reserved for new periodic processes, to processor capacity that is required for the asynchronous process if left unconverted, exceeds a predetermined threshold value. *("Each of [1], [2], and [3] is a condition for improving the theoretical effectiveness, and each of [4] and [5] is a condition for doing the same by determining "n" heuristically, or from experience. Depending on the application which is running, "n" is adjusted in order to improve efficiency. With respect to conditions [4] and [5], instead of the number of processes waiting for synchronization, the ratio of the number of processors in the group to the number of processors waiting for synchronization in the group is used," col. 6, lines 25-35*). It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of synchronizing with respects to a ratio

value of processing capacity for the reason of increasing the control of the system. This ratio tells the processor when it can stop waiting for synchronization to begin, for example. As mentioned earlier, it is common knowledge in the art of task management and process synchronization that converting asynchronous processes to synchronous ones is merely synchronization. In addition, Dave in view of Lindsley, and in further view of Matsumoto fail to explicitly teach using predetermined thresholds to determine a state in change. However, "Official Notice" is taken that both the concept and advantages of providing that the use of thresholds is well known and expected in the art. It would have been obvious to one of ordinary skill in the art at the time the invention was made to include thresholds to the existing method for the reason of increasing the control by being able to set limits or boundaries which determine one state over another. In this specific case, synchronization would begin after the threshold is reached." (Last O.A.)

Applicant respectfully disagrees with the above "Official Notice", and requests that documentary proof be provided, and the data be stated as specifically as possible, and the facts be supported, under M.P.E.P Section 2144.03 and 37 CFR 1.104(d)(2) for the "Office Notice" position that the use of thresholds is well known and expected in the art in the context of claim 126, i.e.,

" A method as defined in claim 124, in which the determining step comprises

(1) calculating a first processor capacity that is required for the asynchronous process if left unconverted,

(2) calculating a second processor capacity which is required to be reserved for the new periodic process,

(3) determining whether the ratio of said first processor capacity to said second processor capacity exceeds a predetermined threshold value.

would have been obvious to one of ordinary skill in the art at the time the invention was made to include thresholds to the existing method for the reason of increasing the control by being able to set limits or boundaries which determine one state over another. In this specific case, synchronization would begin after the threshold is reached."

(a) Matsumoto fails to show any of the important features of claim 126. The quoted phrases, "*of processes waiting for synchronization, the ratio of the number of processors in the group to the number of processors waiting for synchronization in the group is used,*"(Matsumoto *col. 6, lines 25-35*) bear no relation to converting an asynchronous process to a new periodic process. It does not show how to calculate the processor capacity that is required for the asynchronous process if left unconverted, or the processor capacity which is required to be reserved for the new periodic process. Matsumoto fails to show any of the features of claims 115 and 124, on which 126 is based. Matsumoto does not even have the notion of timing constraints, such as periods, worst-case computation times, deadlines, etc.

(b) The art of how to convert an asynchronous process to a new periodic process is not merely "synchronization" as suggested by the last O.A., it is one of the most important, yet least understood, and under-studied techniques in the field of real-time computing. Applicant's papers related to real-time computing have been reprinted in two IEEE Computer Society Tutorial collections and are also widely referenced in textbooks on real-time systems. Applicant is internationally well-known as an expert in real-time computing, and Applicant has taken a special interest in this particular technique for over 20 years, yet it had taken Applicant many, many years before Applicant realized and invented the technique shown by claim 126, hence Applicant can attest to the fact that the technique shown in claim 126 is far, far from obvious.

(c) As can been seen in Fig. 26 of the drawings, and paragraph [0174], determining the ratio of processing capacity of the processor which is required to be reserved for new periodic processes, to processor capacity that is required for the asynchronous process if left unconverted, requires an elaborate procedure that is far from obvious. Hence Applicant is not surprised at all by the fact that no prior art has been found that meets the features shown in claim 126, because, to Applicant's knowledge, up to even today, no one has published a similar invention.

(d) Matsumoto readily acknowledges that his method may include processes that are deadlocked *("All of the processes in the group of processes concerned are dispatched to processors and waiting for synchronization at one time; this event occurs due to programming errors (deadlock)."col. 6, lines 5-9.)* Thus any combination involving Matsumoto will be inoperative, not only because of deadlocks, but in general due to the inability to deal with any timing constraints.

44.4.The Suggested Combination Of Dave, Lindsley, and Matsumoto Would Not Be Physically Possible Or Operative.

There is plenty of unequivocal evidence that show that the suggested combination of Dave, Lindsley, and Matsumoto would not be physically possible or operative, here is just a few of them:

44.4.1. Both Matsumoto and Lindsley describe methods which are totally void of any consideration of quantitative timing constraints.

Matsumoto and Lindsley does not take into account:

(k) worst-case computation time values of asynchronous or periodic processes,

(l) deadline values of asynchronous or periodic processes,

(m) period values of periodic processes

(n) minimum time between two requests of asynchronous processes

(o) permitted range of offset values for periodic processes.

44.4.2. It is notoriously well-known, and common sense would also dictate that, if any single one of the above quantitative timing constraints is not taken into consideration, then there is absolutely no hope of satisfying the timing constraints.

44.4.3. The references of Dave does not contain any teaching, that suggests that their method could be physically combined with the teachings of Lindsley or Matsumoto. Neither does the reference of Lindsley contain any teaching, that

suggests the teaching of Lindsley could be physically combined with the teachings of Dave and Matsumoto, or any other combination could work.

44.4.4. There is no evidence in the prior art that suggests that the references of Dave, Lindsley, and Matsumoto could be combined to produce an operative method.

44.4.5. Combining Dave, Lindsley, and Matsumoto will NOT produce an operative method that will cover all the features of claim 126 as Lindsley and Matsumoto do not concern themselves with timing.

44.4.6. As shown in item 44.1.1 above, neither Lindsley nor Matsumoto do not even consider periodic processes.

44.4.7. As shown in item 44.1.1 above, the last O.A.'s citation of Lindsley is a misunderstood reference. The reference does not teach what the last O.A. relies upon it as supposedly teaching. The main task feature of "synchronous" is not identical to "periodic" as the last O.A. assumed.

44.4.8. The reference of Lindsley relies on the use of general semaphores. It is notoriously well known that general semaphores are not suitable for use in systems with hard deadlines and may prevent the satisfaction of hard deadline constraints altogether. Here is just one simple example: general semaphores are subject to deadlock and various forms of starvation, and just detecting deadlocks alone is an NP-Hard problem, not to mention the total breakdown in any capability to meet any kind of timing constraints when a deadlock does occur. The entire specification of Lindsley completely ignores this problem. As to be expected, the references of Dave and Matsumoto also completely ignore this problem.

44.4.9. Even if Dave is operative when functioning alone, and even if it is structurally possible to combine it with Lindsley, and Matsumoto, the resulting method will NOT be inoperative for the purpose of satisfying hard timing constraints at the very least because of the possibility of deadlock in Lindsley's method.

44.4.10 Actually, Matsumoto readily acknowledges that his method may include processes that are deadlocked *("All of the processes in the group of processes concerned are dispatched to processors and waiting for synchronization at one time; this event occurs due to programming errors (deadlock)."col. 6, lines 5-9.)*

44.4. The Novel Features Of Claim 126 Produce New And Unexpected Results And Hence Are Unobvious And Patentable Over Any Possible Combination Of Dave, Lindsley and Matsumoto Under § 103

Claim 126 provides the following combination of features that have never before been provided simultaneously:

(1) Applicant's invention as defined by claim 126 provides the capability to enforce exclusion relations on pairs of processes that can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are mapped into time slots in the pre-run-time schedule, thus providing the capability to prevent errors caused by more than one process simultaneously accessing shared resources such as data in systems of one or more processors while also providing the capability to select precisely which pairs of process' executions should not overlap in time when they access shared data thus maximizing the system's flexibility in meeting deadline constraints.

Neither Dave, nor Lindsley, nor Matsumoto show this feature.

(2) Applicant's invention as defined by claim 126 simultaneously provides the capability to increase the flexibility of meeting deadline constraints when there is flexibility in assigning an offset value for a periodic process.
Neither Dave, nor Lindsley, nor Matsumoto show this feature.

(3) Applicant's invention as defined by claim 126 provides the above combination of features, while providing the capability to handle processes that can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are mapped into time slots in the pre-run-time schedule, thus obtaining the advantages of pre-run-time scheduling including ability to use the information in the pre-run-time schedule to achieve greater predictability, ability to handle complex constraints, lower run-time overhead, and in general greatly increase the efficiency of scheduling; and the advantages of run-time scheduling including ability to handle asynchronous processes with very short deadlines that cannot be converted into periodic processes or will waste too much processor capacity if converted into periodic processes, while providing a guarantee that all the constraints, including hard deadlines, will be satisfied before run-time.
Neither Dave, nor Lindsley, nor Matsumoto show this feature show this combination of features.

A. Applicant's invention achieves **unexpected results**: A system with all the above important features combined together, has never been realized before. The combination of results achieved by Applicant's invention are new and vastly superior compared to that of Dave, Lindsley, and Matsumoto or any combination thereof.

B. Applicant's invention is classified in a **crowded art** (a prior art patent cited by the O.A. states that, *"There is a vast amount of literature in the area of scheduling of soft and hard aperiodic tasks", Dave, col. 2, lines 47-48*); therefore, even a small step forward should be regarded as significant.

Applicant therefore submits that claim 126 is patentable under § 102 and § 103 and should be allowed, since they produce new and unexpected results over Dave, Lindsley, and Matsumoto or any combination thereof.

45. The Rejection Of Claims 67-68 Under 35 USC § 103 On Dave (US 6,178,542 B1), Dave2 (US 6,086,628), Lindsley (US 6,430,593 B1) And Nilsen (US 6,438,573 B1) Is Overcome

46. The last O.A. rejected dependent claims 67 on Dave, Dave2, Lindsley, and Nilsen. Claims 67 has been rewritten as new dependent claims 121 to define patentably over Dave, Dave2, Lindsley, and Nilsen and any combination thereof.
Applicant submits that claim 121 is independently patentable over Dave, Dave2, Lindsley, and Nilsen and any combination therefore for the same reasons as given in item 13 and item 15 above, and in addition, the following reasons.

(1) There is no justification, in Dave, Dave2, Lindsley, and Nilsen, or in any other prior art separate from Applicant's disclosure, which suggests that these references be combined, much less combined in the manner proposed.

(2) Even if Dave, Dave2, Lindsley, and Nilsen were to be combined in the manner proposed, the proposed combination would not show all of the novel physical features of claim 121.

(3) The proposed combination would not be physically possible or operative.

(4) The novel features of claim 121 produce new and unexpected results and hence are unobvious and patentable over these references.

46.1. Dave, Dave2, Lindsley, and Nilsen Do Not Contain Any Justification To Support Their Combination, Much Less In The Manner Proposed

With regard to the proposed combination of Dave, Dave2, Lindsley, and Nilsen, it is well known that in order for any prior-art references themselves to be validly combined for use in a prior-art § 103 rejection, *the references themselves* (or some other prior art) must suggest that they be combined. E.g., as was stated in In re Sernaker, 217 U.S.P.Q. 1, 6 (C.A.F.C. 1983):

> "[P]rior art references in combination do not make an invention obvious unless something in the prior art references would suggest the advantage to be derived from combining their teachings."

That the suggestion to combine the references should not come from applicant was forcefully stated in Orthopedic Equipment Co. v. United States, 217 U.S.PQ. 193, 199 (CAFC 1983):

> "It is wrong to use the patent in suit [here the patent application] as a guide through the maze of prior art references, combining the right references in the right way to achieve the result of the claims in suit [here the claims pending]. Monday morning quarterbacking is quite proper when resolving the question of nonobviousness in a court of law [here the PTO]."

As was further stated in Uniroyal, Inc. v. Rudkin-Wiley Corp., 5 U.S.P.Q.2d 1434 (C.A.F.C. 1988), "[w]here prior-art references require selective combination by the court to render obvious a subsequent invention, there must be some reason for the combination other than the hindsight gleaned from the invention itself. . . . *Something in the prior art must suggest the desirability and thus the obviousness of making the combination.*" [Emphasis supplied.]

In line with these decisions, the Board stated in Ex Parte Levengood, 28 U.S.P.Q.2d 1300 [P.T.O.B.A.&]. 1993):

"In order to establish a *prima facie* case of obviousness, it is necessary for the examiner to present evidence, preferably in the form of some teaching, suggestion, incentive or inference in the applied prior art, or in the form of generally available knowledge, that one having ordinary skill in the art *would have been led to* combine the relevant teachings of the applied references in the proposed manner to arrive at the claimed invention. . . . That which is within the capabilities of one skilled in the art is not synonymous with obviousness, ... That one can *reconstruct* and/or explain the theoretical mechanism of an invention by means of logic and sound scientific reasoning does not afford the basis for an obviousness conclusion unless that logic and reasoning also supplies sufficient impetus to have led one of ordinary skill in the art to combine the teachings of the references to make the claimed invention. . . . Our reviewing courts have often advised the Patent and Trademark Office that it can satisfy the burden of establishing a *prima facie* case of obviousness only be showing some objective teaching in either the prior art, or knowledge generally available to one of ordinary skill in the art, that 'would lead' that individual 'to combine the relevant teachings of the references." ... Accordingly, an examiner cannot establish obviousness by locating references which describe various aspects of a patent application's invention without also providing evidence of the motivating force which would impel one skilled in the art to do what the patent applicant has done."

In the present case, there is no reason given in the last O.A. to support the proposed combination, other than the statement *"Referring to claim 67, Dave in view of Lindsley teaches using time slicing/slots mapped into a schedule but fails to explicitly teach a method prior to the mapping step, automatically adjusting lengths of periods of a predetermined set of periodic processes, generating a set of reference periods, setting the length of each periodic process to the length of the largest reference period that is no larger than an original period of the periodic process to form adjusted periods, and storing the adjusted periods for subsequent use in pre-run-time scheduling of executions of the periodic processes. However, Nilsen teaches time periods being adjusted ("an ability to adjust the periods of activity tasks", "adjust task periods so that they align more evenly with the current period of the system's real-time cyclic schedule", col. 23,*

lines 16-23). The reference periods are merely used to be able to adjust the time periods and is inherently used. It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of adjustable lengths of time periods to the existing method for the reason of improving synchronization of the schedule because "they align more evenly with the current period of the systems' real-time cyclic schedule."

However, the fact that the words "*an ability to adjust the periods of activity tasks*" (*col. 23, lines 16-23) "adjust task periods so that they align more evenly with the current period of the system's real-time cyclic schedule", (col. 23, lines 16-23)* appear in the reference is not sufficient to gratuitously and selectively combine parts of one reference (Nilsen's real-time programming method) with another reference in order to meet Applicant's novel claimed combination of features.

Applicant therefore submits that combining Dave, Dave2, Lindsley, and Nilsen is not legally justified and is therefore improper. Thus Applicant submits that the rejection on these references is also improper and should be withdrawn.

Applicant respectfully requests, if the claims are again rejected upon any combination of references, that the Examiner include an explanation, in accordance with M.P.E.P. § 706.02, Ex parte Clapp, 27 U.S.P.Q. 972 (P.O.B.A. 1985), and Ex parte Levengood, supra, a "factual basis to support his conclusion that it would have been obvious" to make the combination,

46.2. Even if Dave, Dave2, Lindsley, and Nilsen Were To Be Combined In The Manner Proposed, The Proposed Combination Would Not Show All The Novel Features Of Claim 121.

However even if the combination of Dave, Dave2, Lindsley, and Nilsen were justified, claim 121 would still have novel and unobvious features over the proposed combination.

46.2.1. Neither Dave, nor Dave2, nor Lindsey, nor Nilsen, nor any possible combination thereof show the feature of **prior to the mapping step, automatically adjusting lengths of periods of a predetermined set of periodic processes, generating a set of reference periods, setting the length of the period of each periodic process to the length of the largest reference period that is no larger than an original period of the periodic process to form adjusted periods, and storing the adjusted periods for subsequent use in pre-run-time scheduling of executions of the periodic processes such that the predetermined constraints will be satisfied.**

Claim 121 clearly distinguishes over Dave, Dave2, Lindsley, and Nilsen since claim 121 includes all the limitations of claim 115 and claim 115 recites:

"*scheduling on one or more processors, executions of a plurality of periodic and asynchronous processes, comprising:*

(A)

automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(1) worst-case computation times for periodic processes and asynchronous processes,

(2) period for periodic processes,

(3) minimum time between two consecutive requests for asynchronous processes,

(4) deadline for periodic processes and asynchronous processes,

(5) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower

bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(6) precedence relations for periodic processes wherein each precedence relation being defined between a pair of processes comprising a first process and a second process, both said first process and said second process being periodic processes, said first process precedes said second process, execution of said second process only allowed to start after said first process has completed its execution,

(7) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots, including the step of converting one or more asynchronous processes into corresponding new periodic processes prior to the mapping step, and mapping new periodic processes to time slots in a manner similar to mapping of other periodic processes, such that said predetermined constraints will be satisfied

(B)

during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such that said predetermined constraints will be satisfied."

Claim 121 further recites:

"*prior to the mapping step, automatically adjusting lengths of periods of a predetermined set of periodic processes, generating a set of reference periods, setting the length of the period of each periodic process to the length of the largest reference period that is no*

larger than an original period of the periodic process to form adjusted periods, and storing the adjusted periods for subsequent use in pre-run-time scheduling of executions of the periodic processes."

Neither Dave, nor Dave2, nor Lindsley, nor Nilsen teach this.

46.2.1.(a) Nilsen merely mentions adjusting periods, but does not offer any specific method on how to adjust periods in a context that is the same as Applicant's invention as defined in claim 121, that is, while guaranteeing that all timing constraints will be satisfied in the system. While Applicant's invention as defined in claim 121 adjusts periods before run-time and guarantees that all specified timing constraints will be satisfied. Nilsen adjusts periods during run-time and does not guarantee that all specified timing constraints will be satisfied. In fact, adjusting the periods in Nilsen's method may cause the system to run behind schedule. ("In these cases, the resource negotiator may choose to adjust task periods so that they align more evenly with the current period of the system's real-time cyclic schedule." col. 23, lines 20-23.) ("Note that the real-time executive may need to revise the time budget dynamically (if, for example, the system finds itself running behind schedule.) Fig. 13, lines -5 to -4.)

46.2.1.(b) Neither Dave, nor Dave2, nor Lindsley, nor Nilsen teach *"generating a set of reference periods,"*

46.2.1.(c) Neither Dave, nor Dave2, nor Lindsley, nor Nilsen teach *" setting the length of the period of each periodic process to the length of the largest reference period that is no larger than an original period of the periodic process to form adjusted periods",*

46.2.1.(d) Neither Dave, nor Dave2, nor Lindsley, nor Nilsen teach *storing the adjusted periods for subsequent use in pre-run-time scheduling of executions of the periodic processes.*"

46.2.1.(e) Nilsen merely mentions using the current period to adjust periods *("In these cases, the resource negotiator may choose to adjust task periods so that they align more evenly with the current period of the system's real-time cyclic schedule." Nilsen, col. 23, lines 20-23.)* but does not give any specific method on how to using the current

period to adjust periods. In contrast, Applicant's invention provides a specific method for adjusting periods, and is not restricted to using only one possibility, the current period.

46.2.1.(f) Nilsen may terminate, abort, or suspend a task in order to guarantee that a task does not exceed a time slot. *("A difficulty arises, however, because the real-time executive desires to control exactly when a particular* task is terminated *in order to prevent one task's sin (i.e., exceeding its allotted time slot from corrupting the integrity of the entire system." Nilsen, col. 9, lines 5-15). ("For a timed-statement control structure, it is anticipated that specified code segment will execute within the specified increment of time, execution otherwise being aborted." Nilsen, col. 3, lines 44-47.) ("Make sure there is sufficient time in the current time slice to execute the complete atomic statement. ... If this condition cannot be satisfied at the current time, suspend the task." Nilsen, col. 17, lines 53-54).* Hence Nilsen does NOT satisfy the predetermined constraints including deadline constraints.

46.2.2. Neither Dave, nor Dave2, nor Lindsley nor Nilsen, nor any possible combination thereof show the feature of automatically generating a pre-run-time schedule in which predetermined constraints comprising worst-case computation time, period, minimum time between two consecutive exclusion relations, deadline, permitted range of offset constraints, precedence relations are satisfied and during run-time executing the processes such that the predetermined constraints are satisfied. (Discussion on Dave's lack of ability to satisfy exclusion constraints is provided in item 11.1. above and is equally applicable to Dave and Dave2 and is hereby included here by reference.)

Claim 121 clearly distinguishes over Dave, Dave2, Lindsley, and Nilsen, or any possible combination thereof, since it is dependent on claim 115 which recites *"automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end*

time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(1) worst-case computation times for periodic processes and asynchronous processes,

(2) period for periodic processes,

(3) minimum time between two consecutive requests for asynchronous processes,

(4) deadline for periodic processes and asynchronous processes,

(5) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(6) precedence relations for periodic processes wherein each precedence relation being defined between a pair of processes comprising a first process and a second process, both said first process and said second process being periodic processes, said first process precedes said second process, execution of said second process only allowed to start after said first process has completed its execution,

(7) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots,

including the step of converting one or more asynchronous processes into corresponding new periodic processes prior to the mapping step, and mapping new periodic processes to time slots in a manner similar to mapping of other periodic processes, such that said predetermined constraints will be satisfied".

Neither Dave, nor Dave2, nor Lindsley, nor Nilsen, nor any possible combination thereof, show the feature of *automatically generating a pre-run-time schedule including satisfaction of predetermined constraints comprising*
"exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation".

46.2.2 (a) Nilsen does not enforce exclusion constraints. ("It does not enforce mutual exclusion." col. 27, line 5-6).
46.2.2 (b) Neither Lindsley nor Nilsen show "automatically generating a pre-run-time schedule".
46.2.2 (c) Neither Lindsley nor Nilsen does not show "automatically generating a pre-run-time schedule in which predetermined constraints comprising worst-case computation time, period, minimum time between two consecutive requests, deadline, permitted range of offset constraints, precedence relations are satisfied and during run-time executing the processes such that the predetermined constraints are satisfied."
46.2.2.(d) Neither Dave nor Dave2 satisfy exclusion constraints. (This has been shown for Dave in item 11.1. and is equally applicable to Dave and Dave2.)

46.2.2. **Neither Dave, nor Dave2, nor Lindsley, nor Nilsen, nor any possible combination thereof show the feature of permitted range of offset constraints for periodic processes**

Claim 121 which is dependent on claim 115 clearly distinguishes over any combination of Dave, Dave2, Lindsley, and Nilsen since claim 115 recites

"including satisfaction of predetermined constraints comprising

...

(4) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,"

Dave, nor Dave2, nor Lindsley, nor Nilsen, nor any possible combination thereof show the feature of satisfying predetermined constraints comprising "*permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,*"

46.3. The Suggested Combination Of Dave, Dave2, Lindsley, and Nilsen Would Not Be Physically Possible Or Operative.

There is plenty of unequivocal evidence that show that the suggested combination of Dave, Dave2, Lindsley, and Nilsen would not be physically possible or operative, here is just a few of them:

46.3.1. Nilsen does NOT provide any guarantee, either before run-time, nor during run-time that all specified timing constraints, including deadline constraints, will be satisfied:

Nilsen explicitly acknowledges that it is possible that the system's timing is not guaranteed:

(a) The system may run behind schedule. *("Note that the real-time executive may need to revise the time budget dynamically (if, for example, the system finds itself running behind schedule.),"* Fig. 13, lines -5 to -4.)

(b) Time alarms may be postponed indefinitely. *("If the alarm was set before entering into the current atomic segment and the alarm time arrives while the application is still executing the body of the atomic segment, delivery of the alarm's exception is postponed until after the application leaves its atomic statement." col. 18, lines 41-45.)*

(c) Tasks can be terminated, aborted, and suspended at any time. *("A difficulty arises, however, because the real-time executive desires to control exactly when a particular* task is terminated *in order to prevent one task's sin (i.e., exceeding its allotted time slot from corrupting the integrity of the entire system." Nilsen, col. 9, lines 5-15). ("For a timed-statement control structure, it is anticipated that specified code segment will execute within the specified increment of time, execution otherwise being aborted." Nilsen, col. 3, lines 44-47.) ("Make sure there is sufficient time in the current time slice to execute the complete atomic statement. ... If this condition cannot be satisfied at the current time, suspend the task." Nilsen, col. 17, lines 53-54).*

(d) Nilsen's method is not intended to result in a system such that all predetermined constraints including deadline constraints are satisfied

("The large majority of code comprising an RTPM program is not intended to be execution time analyzable." Nilsen, col. 6, lines 27-28.)

(e) Some code segments of a task may never be assigned time to allow them to be executed. *("The RTPM also utilizes an atomic-statement control structure. ... the control structure requiring that either enough time be assigned to execute the specified code segment or that none be assigned." Nilsen, col. 3, lines 58-64.)*

46.3.2. Lindsley describes a method which is totally void of any consideration of quantitative timing constraints.

Lindsley does not take into account:

(a) worst-case computation time values of asynchronous or periodic processes,

(b) deadline values of asynchronous or periodic processes,

(c) period values of periodic processes

(d) minimum time between two requests of asynchronous processes

(e) permitted range of offset values for periodic processes.

46.3.3. The references of Dave and Dave2 do not contain any teaching, that suggests that their method could be physically combined with the teachings of Lindsley or Nilsen. Neither does the references of Lindsley or Nilsen contain any teaching, that suggests the teaching of Lindsley or Nilsen could be physically combined with the teachings of Dave and Dave2.

46.3.4. There is no evidence in the prior art that suggests that the references of Dave, Dave2, Lindsley, and Nilsen could be combined to produce an operative method. Dave and Dave2 are off-line co-synthesis algorithms that do not schedule processes during run-time, while Lindsley and Nilsen have no knowledge of pre-run-time schedules, and timing constraints in general.

46.3.5. Combining Dave, Dave2, Lindsley, and Nilsen will NOT produce an operative method that will cover all the features of claim 121 as neither Lindsley nor Nilsen guarantee that predetermined constraints including deadlines will be satisfied.

46.3.6. Even if Dave and Dave2, are operative when functioning alone, and even if it is structurally possible to combine them with Lindsley or Nilsen, the resulting method will be inoperative for the purpose of satisfying hard timing constraints at the very least because of the possibility of terminating, or aborting tasks before the tasks are completed, and the general inability to guarantee that timing constraints including deadline constraints will be satisfied in Nilsen's and Lindsley's method.

The above evidence, clearly shows that the O.A.'s reason for combining Dave, Dave2, Lindsley, and Nilsen to find obviousness of claim 121, i.e., "*an ability to adjust the periods of activity tasks*" (*col. 23, lines 16-23) "adjust task periods so that they align more evenly with the current period of the system's real-time cyclic schedule", (col. 23, lines 16-23)* is not a valid reason.

46.4. The Novel Features Of Claim 121 Produce New And Unexpected Results And Hence Are Unobvious And Patentable Over Any Possible Combination Of Dave, Dave 2, Lindsley, and Nilsen Under § 103

Claim 121 provides the following combination of features that have never before been provided simultaneously:

(1) Applicant's invention as defined by claim 121 provides the capability to enforce exclusion relations on pairs of processes that can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous

processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are mapped into time slots in the pre-run-time schedule, thus providing the capability to prevent errors caused by more than one process simultaneously accessing shared resources such as data in systems of one or more processors while also providing the capability to select precisely which pairs of process' executions should not overlap in time when they access shared data thus maximizing the system's flexibility in meeting deadline constraints.

Neither Dave, nor Dave 2, nor Lindsley, nor Nilsen show this feature.

(2) Applicant's invention as defined by claim 121 simultaneously provides the capability to increase the flexibility of meeting deadline constraints when there is flexibility in assigning an offset value for a periodic process.

Neither Dave, nor Dave 2, nor Lindsley, nor Nilsen show this feature.

(3) Applicant's invention as defined by claim 121 simultaneously provides the capability to reduce the length of the pre-run-time schedule by adjusting the length of the periods while simultaneously satisfying all the predetermined constraints mentioned above.

Neither Dave, nor Dave 2, nor Lindsley, nor Nilsen show this feature.

(4) Applicant's invention as defined by claim 121 provides the above combination of features, while providing the capability to handle processes that can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are mapped into time slots in the pre-run-time schedule, thus obtaining the advantages of pre-run-time scheduling including ability to use the information in the pre-run-time schedule to achieve greater predictability, ability to handle complex constraints, lower run-time overhead, and in general greatly increase the efficiency of scheduling; and the advantages of run-time scheduling including ability to handle asynchronous processes with very short deadlines that cannot be converted into periodic processes or will waste too much processor capacity if

converted into periodic processes, while providing a guarantee that all the constraints, including hard deadlines, will be satisfied before run-time.
Neither Dave, nor Dave 2, nor Lindsley, nor Nilsen show this combination of features.

A. Applicant's invention achieves **unexpected results**: A system with all the above important features combined together, has never been realized before. The combination of results achieved by Applicant's invention are new and vastly superior compared to that of Dave, Dave2, Lindsley, and Nilsen or any combination thereof.

B. Applicant's invention is classified in a **crowded art** (a prior art patent cited by the O.A. states that, *"There is a vast amount of literature in the area of scheduling of soft and hard aperiodic tasks", Dave, col. 2, lines 47-48*); therefore, even a small step forward should be regarded as significant.

Applicant therefore submits that claim 121 is patentable under § 102 and § 103 and should be allowed, since they produce new and unexpected results over Dave, Dave2, Lindsley, and Nilsen or any combination thereof.

47. The last O.A. rejected dependent claims 68 on Dave, Dave2, Lindsley, and Nilsen. Claims 68 has been rewritten as new dependent claims 122 to define patentably over Dave, Dave2, Lindsley, and Nilsen and any combination thereof.
Applicant submits that claim 122 is independently patentable over Dave, Dave2, Lindsley, and Nilsen and any combination.
Applicant requests reconsideration of this rejection, as now applicable to claim 122, for the same reasons as given in item 13, item 15, item 46, and item 52, and in addition, the following reasons:

Claim 122 clearly distinguishes over Dave, Dave2, Lindsley, and Nilsen since it recites:

"prior to the mapping step, automatically adjusting lengths of periods of a predetermined set of periodic processes by generating a list of reference periods, setting the length of the period of each periodic process to the length of the largest reference period that is no larger than an original period of the periodic process to form adjusted periods, and storing the adjusted periods for subsequent use in pre-run-time scheduling of executions of the periodic processes."

Neither Dave, nor Dave2, nor Lindsley, nor Nilsen do this. In addition to the reasons given in item 46, and further in addition to the reasons given in item 27 and item 30 of this amendment, note that:

28.1. Nilsen only uses the current period to adjust periods *("In these cases, the resource negotiator may choose to adjust task periods so that they align more evenly with the current period of the system's real-time cyclic schedule." Nilsen, col. 23, lines 20-23.)*

28.2. Neither Dave, nor Dave2, nor Lindsley, nor Nilsen generate a list of reference periods. Neither Dave, nor Dave2, nor Lindsley, nor Nilsen store the adjusted periods for subsequent use in pre-run-time scheduling of executions of the periods of the periodic processes.

28.3. In addition, neither Dave, nor Dave2, nor Lindsley, nor Nilsen adjust periods while satisfying the predetermined constraints (1)-(7) defined in claim 115 (all the limitations of claim 115 are included in claim 122).

48. The Rejection Of Claims 80 And 111 Under 35 USC § 103 On Dave (US 6,178,542 B1), Dave2 (US 6,086,628), Lindsley (US 6,430,593)*, and Fong (US 6,345,287 B1) Is Overcome

49. "Applicant would like to draw attention to a possible mistake in the last O.A, page 20, item 48-49: The last O.A. rejected dependent claim 80 on Dave, Dave2, and Fong; and only Dave, Dave2, and Fong were mentioned in the last O.A. comments related to claim 80. But the last O.A. comments also said that "Referring to claim 80, it rejected for

the same reasons as stated in the rejection of claims 59,-64, 75, and 78." In the last O.A., page 8, item 14, claims 59,-64, 75, and 78 were rejected on Dave, Dave2, and Lindsley. Since Applicant is not sure whether to write a response based on (a) the assumption that claim 80 was rejected on Dave, Dave2, and Fong; or, (b) the assumption that claim 80 was rejected on Dave, Dave2, Lindsley, and Fong. In this amendment, Applicant will write a response base on the latter assumption, i.e. (b) the assumption that claim 80 was rejected on Dave, Dave2, Lindsley, and Fong. However, Applicant notes that the Applicant's response would have been different if assumption (a) was made.

Claim 80 has been rewritten as new independent claim 130 to define patentably over Dave, Dave2, Lindsley, and Fong and any combination thereof. Applicant requests reconsideration of this rejection, as now applicable to claim 130, for all the reasons as stated in item 13 and item 15 of this amendment, and in addition, for the following reasons:

(1) There is no justification, in Dave, Dave2, Lindsley, and Fong, or in any other prior art separate from Applicant's disclosure, which suggests that these references be combined, much less combined in the manner proposed.
(2) The proposed combination would not be physically possible or operative.
(3) Even if Dave, Dave2, Lindsley, and Fong were to be combined in the manner proposed, the proposed combination would not show all of the novel physical features of claim 130.
(4) The novel features of claim 130 produce new and unexpected results and hence are unobvious and patentable over these references.

49.1. Dave, Dave2, Lindsley, and Fong Do Not Contain Any Justification To Support Their Combination, Much Less In The Manner Proposed

With regard to the proposed combination of Dave, Dave2, Lindsley, and Fong, it is well known that in order for any prior-art references themselves to be validly combined for use in a prior-art § 103 rejection, *the references themselves* (or some other prior art) must

suggest that they be combined. E.g., as was stated in In re Sernaker, 217 U.S.P.Q. 1, 6 (C.A.F.C. 1983):

> "[P]rior art references in combination do not make an invention obvious unless something in the prior art references would suggest the advantage to be derived from combining their teachings."

That the suggestion to combine the references should not come from applicant was forcefully stated in Orthopedic Equipment Co. v. United States, 217 U.S.PQ. 193, 199 (CAFC 1983):

> "It is wrong to use the patent in suit [here the patent application] as a guide through the maze of prior art references, combining the right references in the right way to achieve the result of the claims in suit [here the claims pending]. Monday morning quarterbacking is quite proper when resolving the question of nonobviousness in a court of law [here the PTO]."

As was further stated in Uniroyal, Inc. v. Rudkin-Wiley Corp., 5 U.S.P.Q.2d 1434 (C.A.F.C. 1988), "[w]here prior-art references require selective combination by the court to render obvious a subsequent invention, there must be some reason for the combination other than the hindsight gleaned from the invention itself. . . . *Something in the prior art must suggest the desirability and thus the obviousness of making the combination.*" [Emphasis supplied.]

In line with these decisions, the Board stated in Ex Parte Levengood, 28 U.S.P.Q.2d 1300 [P.T.O.B.A.&]. 1993):

"In order to establish a *prima facie* case of obviousness, it is necessary for the examiner to present evidence, preferably in the form of some teaching, suggestion, incentive or inference in the applied prior art, or in the form of generally available knowledge, that one having ordinary skill in the art *would have been led to* combine the

claimed invention. . . . That which is within the capabilities of one skilled in the art is not synonymous with obviousness, ... That one can *reconstruct* and/or explain the theoretical mechanism of an invention by means of logic and sound scientific reasoning does not afford the basis for an obviousness conclusion unless that logic and reasoning also supplies sufficient impetus to have led one of ordinary skill in the art to combine the teachings of the references to make the claimed invention. . . . Our reviewing courts have often advised the Patent and Trademark Office that it can satisfy the burden of establishing a *prima facie* case of obviousness only be showing some objective teaching in either the prior art, or knowledge generally available to one of ordinary skill in the art, that 'would lead' that individual 'to combine the relevant teachings of the references." ... Accordingly, an examiner cannot establish obviousness by locating references which describe various aspects of a patent application's invention without also providing evidence of the motivating force which would impel one skilled in the art to do what the patent applicant has done."

In the present case, there is no reason given in the last O.A. to support the proposed combination, other than the statement *"Referring to claim 80, it is rejected for the same reasons as stated in the rejection of claims 59-64, 75, and 78. Dave in view of Dave2 fails to explicitly teach scheduling with subschedules. However, Fong teaches using subpartitions and subschedules ("subpartitions", "subschedules", col. 3, lines 6-32, and col. 3, lines 55-67, and col. 4, lines 46-65). It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of subschedules/subpartitions to the existing method for the reason of increasing the flexibility of the system (When only a mapping of applications to processors is provided by the higher level scheduler, there is additional flexibility and generality by allowing the lower level schedulers to make all or any subset of the scheduling decisions.", col. 3, lines 7-32.*

However, the fact that the words "subpartitions", "subschedules" (Fong, *col. 3, lines 6-32, and col. 3, lines 55-67, and col. 4, lines 46-65).* appear in the reference is not sufficient to gratuitously and selectively combine parts of one reference (Fong's gang

scheduling) with another reference in order to meet Applicant's novel claimed combination of features.

Note that:

49.1.1. The meaning of the term "synchronous" in Lindsley is inherently different from the meaning of "periodic" in Applicant's invention as defined by claim 130 as shown below:

49.1.1.(a) Lindsley uses the term "synchronous" when describing "synchronous task commands". *("The TSA accepts commands from tasks called `synchronous' task commands. These commands are synchronous from the point of view that the task may not issue another synchronous task command until the previous synchronous task command has been completed. It is allowable for the task to perform other activity after issuing a synchronous task command as long as the task verifies that the previous task has been completed prior to issuing another synchronous task command." Fong, col. 6, lines 47-55.)* It is clear that in Lindsley there is no periodic constraint on "synchronous" task commands, i.e., synchronous task commands are not constrained to execute strictly once in each fixed period of time.

49.1.1.(b) In contrast, in Applicant's invention as defined by claim 130, "a periodic process consists of a computation that is executed repeatedly, once in each fixed period of time. (Applicant's specification, paragraph [0113]). Applicant's definition of a periodic process, is adopted universally in the field of real-time computing and is clearly different from the meaning of "synchronous" as defined in Fong.

49.1.2. The meaning of the term "*synchronization*" in Lindsley is also inherently different from the meaning of "*converting an asynchronous process to a new periodic process*" in Applicant's invention as defined by claim 130 as shown below:

49.1.2.(a) In Lindsley, tasks are synchronized using "events" which are certainly not always periodic.*("Tasks are typically synchronized using "events". An event is used to indicate that an activity has taken place, such as data arrival, time-out, etc. Thus, an event may indicate execution of a task, an interrupt service routine or the like. Events are*

counted using semaphores. Semaphores synchronize the event producer and the event consumer ... " Fong, col. 2, lines 4-10.)

("If a task that processes data buffers pends for the semaphore that represents data buffers, the task is synchronized to the data buffer generation. If data buffers are available, the semaphore count is greater than zero. Task pend requests on the semaphore allow the task to continue. If data buffers are not available, the semaphore count is less than or equal to zero, the task does not have data for processing and will block execution". Lindsley, col. 2, lines 52-61.)

49.1.2.(b) In contrast, in Applicant's invention as defined by claim 130, "*converting an asynchronous process to a new periodic process*" means converting a process that can be executed at random times, to a process that will be executed once in each fixed period of time.

The above evidence, clearly shows that this O.A.'s reason for combining Dave, Dave2, Lindsley, and Fong to find obviousness of claim 130, i.e., *("It is common knowledge in the art of task management that converting an asynchronous process to a new periodic process (or a synchronous process) is known as synchronization,"* is totally unfounded.

49.1.3.(a) Neither Dave, nor Dave2, nor Lindsley, nor Fong say that their methods can be combined with other scheduling methods. This is a good indication that neither Dave, nor Dave2, nor Lindsley, nor Fong can integrate their methods with another methods, because, the capability to be integrated with other scheduling methods is a valuable and marketable advantage that an inventor will never fail to claim if that capability indeed exists.

49.1.3.(b) Neither Dave, nor Dave2, nor Lindsley, nor Fong teach the method of determining the point that separates the initial part and the repeating part of the pre-run-time schedule in claim 130.

legally justified and is therefore improper. Thus Applicant submits that the rejection on these references is also improper and should be withdrawn.

Applicant respectfully requests, if the claims are again rejected upon any combination of references, that the Examiner include an explanation, in accordance with M.P.E.P. § 706.02, Ex parte Clapp, 27 U.S.P.Q. 972 (P.O.B.A. 1985), and Ex parte Levengood, supra, a "factual basis to support his conclusion that it would have been obvious" to make the combination,

49.2. Even if Dave, Dave2, Lindsley, and Fong Were To Be Combined In The Manner Proposed, The Proposed Combination Would Not Show All The Novel Features Of Claim 130.

However even if the combination of Dave, Dave2, Lindsley, and Fong were justified, claim 130 would still have novel and unobvious features over the proposed combination.

49.2.1. Neither Dave, nor Dave2, nor Fong, nor any possible combination thereof show the feature of automatically generating a pre-run-time schedule in which predetermined constraints comprising worst-case computation time, period, minimum time between two consecutive exclusion relations, deadline, permitted range of offset constraints, precedence relations are satisfied. (Discussion on Dave's lack of ability to satisfy exclusion constraints is provided in item 11.1. above and is equally applicable to Dave and Dave2 and is hereby included here by reference.)

Claim 130 clearly distinguishes over Dave, Dave2, Lindsley, and Fong, or any possible combination thereof, since it is dependent on claim 115 which recites *"automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time*

axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(8) worst-case computation times for periodic processes and asynchronous processes,

(9) period for periodic processes,

(10) minimum time between two consecutive requests for asynchronous processes,

(11) deadline for periodic processes and asynchronous processes,

(12) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(13) precedence relations for periodic processes wherein each precedence relation being defined between a pair of processes comprising a first process and a second process, both said first process and said second process being periodic processes, said first process precedes said second process, execution of said second process only allowed to start after said first process has completed its execution,

(14) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to

occur between the time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots, including the step of converting one or more asynchronous processes into corresponding new periodic processes prior to the mapping step, and mapping new periodic processes to time slots in a manner similar to mapping of other periodic processes, such that said predetermined constraints will be satisfied".

Neither Dave, nor Dave2, nor Lindsley, nor Fong, nor any possible combination thereof, show the feature of *automatically generating a pre-run-time schedule including satisfaction of predetermined constraints comprising "exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation".*
(Discussion on Dave's lack of this feature is provided in item 11.1. above and is equally applicable to Dave and Dave2, and is hereby included here by reference.)

Note that Lindsley performs synchronization only at run-time, NOT before run-time, so it is not capable of satisfying any constraints before run-time, which Applicant's invention, as defined by claim 130 would be capable of doing. Fong does not show any feature related to exclusion relations, consequently it does not show the feature above.

49.2.2. Neither Dave, nor Dave2, nor Lindsley, nor Fong, nor any possible combination thereof show the feature of permitted range of offset constraints for periodic processes

Claim 130 which is dependent on claim 115 clearly distinguishes over any combination of Dave, Dave2, Lindsley, and Fong since claim 115 recites

"including satisfaction of predetermined constraints comprising

...

(4) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value, n,"

Dave, nor Dave2, nor Lindsley, nor Fong, nor any possible combination thereof show the feature of satisfying predetermined constraints comprising "*permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value, n,*"

49.2.2. Neither Dave, nor Dave2, nor Lindsley, nor Fong, nor any possible combination thereof show the feature of determining the point that separates the initial part and the repeating part of the two-part pre-run-time schedule and how to construct the two-part pre-run-time schedule.

Claim 130 clearly distinguishes over any combination of Dave, Dave2, Lindsley, and Fong since claim 130 recites

"constructing a first schedule for executions of the periodic processes within an interval starting from zero and having length equal to maximum offset value plus

a bounded number of times of the length of a least common multiple of the periodic process periods, conditions for determining feasibility requiring the existence of a point in said first schedule wherein starting from the latter point the schedule repeats in subschedule interval lengths equal to a least common multiple of lengths of the periodic process periods, timing of all executions of all periodic processes within a time interval having length equal to the length of the least common multiple of the periodic process periods being included in each said repeating subschedule interval, and including satisfaction of all predetermined constraints for all executions of all periodic processes within the subschedule interval starting from time zero and ending at said point plus the length of the least common multiple of the periodic process periods in said first schedule, and checking for the first occurrence of said point in said first schedule,

(B)

generating said feasible two-part pre-run-time-schedule by

(1) using a subschedule interval starting from time zero and ending at said point in said first schedule as said initial part of said feasible two-part pre-run-time schedule, and

(2) using a subschedule interval starting from said point and ending at said point plus the length of the least common multiple of the periodic process periods in said first schedule as said repeating part of said feasible two-part pre-run-time schedule.

Dave, nor Dave2, nor Lindsley, nor Fong, nor any possible combination thereof show the feature of determining the point that separates the initial part and the repeating part of the two-part pre-run-time schedule and how to construct the two-part pre-run-time schedule.

49.2.3. Neither Dave, nor Dave2, nor Lindsley, nor Fong, nor any possible combination thereof show the features of claim 127 which are incorporated in claim 130

Claim 130 clearly distinguishes over Dave, Dave2, Lindsley, and Fong since claim 130 includes all the limitations of claim 127, and claim 127 recites:

"*generating the pre-run-time schedule as a feasible two-part pre-run-time-schedule for execution of periodic processes that may have non-zero offsets (a) an initial part which may be of zero length, and (b) a repeating part having length which is equal to a least common multiple of lengths of the periods of the periodic processes,*

all executions of all periodic processes within a time interval of length equal to the length of the least common multiple of the periodic process periods being included in the repeating part of the pre-run-time schedule, wherein all said predetermined constraints being satisfied for all executions of all periodic processes within said initial part and said repeating part."

Claim 130 further recites:

"*the steps of*

(A)

constructing a first schedule for executions of the periodic processes within an interval starting from zero and having length equal to maximum offset value plus a bounded number of times of the length of a least common multiple of the periodic process periods, conditions for determining feasibility requiring the existence of a point in said first schedule wherein starting from the latter point the schedule repeats in subschedule interval lengths equal to a least common multiple of lengths of the periodic process periods, timing of all executions of all periodic processes within a time interval having length equal to the length of the least common multiple of the periodic process periods being included in each said repeating subschedule interval, and including satisfaction of all predetermined constraints for all executions of all periodic processes within the subschedule interval starting from time zero and ending at said point plus the length of the least common multiple of the periodic process periods in said first schedule, and checking for the first occurrence of said point in said first schedule,

(B)

generating said feasible two-part pre-run-time-schedule by

(1) using a subschedule interval starting from time zero and ending at said point in said first schedule as said initial part of said feasible two-part pre-run-time schedule, and
(2) using a subschedule interval starting from said point and ending at said point plus the length of the least common multiple of the periodic process periods in said first schedule as said repeating part of said feasible two-part pre-run-time schedule."

Neither Dave, nor Dave2, nor Lindsley, nor Fong do this. Dave and Dave2 only use a single repeating schedule. Lindsley and Fong have no knowledge of quantitative timing constraints.

49.3. The Suggested Combination Of Dave, Dave2, Lindsley, and Fong Would Not Be Physically Possible Or Operative.

There is plenty of unequivocal evidence that show that the suggested combination of Dave, Dave2, Lindsley, and Fong would not be physically possible or operative, here is just a few of them:

49.3.1 Lindsley and Fong describe methods which is totally void of any consideration of quantitative timing constraints.

Lindsley and Fong does not take into account:

(a) worst-case computation time values of asynchronous or periodic processes,
(b) deadline values of asynchronous or periodic processes,
(c) period values of periodic processes
(d) minimum time between two requests of asynchronous processes
(e) permitted range of offset values for periodic processes.

49.3.2. It is notoriously well-known, and common sense would also dictate that, if any single one of the above quantitative timing constraints is not taken into consideration, then there is absolutely no hope of satisfying the timing constraints.

49.3.3. The references of Dave and Dave2 do not contain any teaching, that suggests that their method could be physically combined with the teachings of Lindsley or Fong. Neither does the reference of Lindsley, or Fong contain any teaching, that suggests the teaching of Lindsley, or Fong could be physically combined with each other, or with the teachings of Dave and Dave2.

49.3.4. There is no evidence in the prior art that suggests that the references of Dave, Dave2, Lindsley, and Fong could be combined to produce an operative method.

49.3.5. Combining Dave, Dave2, Lindsley, and Fong will NOT produce an operative method that will cover all the features of claim 130 as Lindsley, and Fong do not concern themselves with timing.

49.3.6. As shown in item 49.1.1 above, Lindsley does not even consider periodic processes. This is also true for Fong.

49.3.7. As shown in item 49.1.1 above, the last O.A.'s citation of Lindsley is a misunderstood reference. The reference does not teach what the last O.A. relies upon it as supposedly teaching. The main task feature of "synchronous" is not identical to "periodic" as the last O.A. assumed.

49.3.8. The reference of Lindsley relies on the use of general semaphores. It is notoriously well known that general semaphores are not suitable for use in systems with hard deadlines and may prevent the satisfaction of hard deadline constraints altogether. Here is just one simple example: general semaphores are subject to deadlock and various forms of starvation, and just detecting deadlocks

alone is an NP-Hard problem, not to mention the total breakdown in any capability to meet any kind of timing constraints when a deadlock does occur. The entire specification of Lindsley completely ignores this problem. As to be expected, the references of Dave, Dave2, and Fong also completely ignores this problem.

49.3.9. Even if Dave and Dave2, are operative when functioning alone, and even if it is structurally possible to combine them with Fong, or Lindsley, the resulting method will be inoperative for the purpose of satisfying hard timing constraints at the very least because of the possibility of deadlock in Lindsley's method.

49.4. The Novel Features Of Claim 130 Produce New And Unexpected Results And Hence Are Unobvious And Patentable Over Any Possible Combination Of Dave, Dave 2, Lindsley, and Fong Under § 103

Claim 130 provides the following combination of features that have never before been provided simultaneously:

(1) Applicant's invention as defined by claim 130 provides the capability to enforce exclusion relations on pairs of processes that can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are mapped into time slots in the pre-run-time schedule, thus providing the capability to prevent errors caused by more than one process simultaneously accessing shared resources such as data in systems of one or more processors while also providing the capability to select precisely which pairs of process' executions

should not overlap in time when they access shared data thus maximizing the system's flexibility in meeting deadline constraints.
Neither Dave, nor Dave 2, nor Fong show this feature.

(2) Applicant's invention as defined by claim 130 simultaneously provides the capability to increase the flexibility of meeting deadline constraints when there is flexibility in assigning an offset value for a periodic process.
Neither Dave, nor Dave 2, nor Fong show this feature.

(3) Applicant's invention as defined by claim 130 simultaneously provides the capability to determining the point that separates the initial part and the repeating part of the two-part pre-run-time schedule and how to construct the two-part pre-run-time schedule that satisfies all specified constraints.
Neither Dave, nor Dave 2, nor Fong show this feature.

(3) Applicant's invention as defined by claim 130 provides the above combination of features, while providing the capability to handle processes that can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are mapped into time slots in the pre-run-time schedule, thus obtaining the advantages of pre-run-time scheduling including ability to use the information in the pre-run-time schedule to achieve greater predictability, ability to handle complex constraints, lower run-time overhead, and in general greatly increase the efficiency of scheduling; and the advantages of run-time scheduling including ability to handle asynchronous processes with very short deadlines that cannot be converted into periodic processes or will waste too much processor capacity if converted into periodic processes, while providing a guarantee that all the constraints, including hard deadlines, will be satisfied before run-time.
Neither Dave, nor Dave 2, nor Fong show this combination of features.

A. Applicant's invention achieves **unexpected results**: A system with all the above important features combined together, has never been realized before. The combination of results achieved by Applicant's invention are new and vastly superior compared to that of Dave, Dave2, Lindsley, and Fong or any combination thereof.

B. Applicant's invention is classified in a **crowded art** (a prior art patent cited by the O.A. states that, *"There is a vast amount of literature in the area of scheduling of soft and hard aperiodic tasks", Dave, col. 2, lines 47-48*); therefore, even a small step forward should be regarded as significant.

Applicant therefore submits that claim 130 is patentable under § 102 and § 103 and should be allowed, since they produce new and unexpected results over Dave, Dave2, Lindsley, and Fong or any combination thereof.

50. The last O.A. rejected dependent claim 111 on Dave, Dave2, (*Lindsley, see comments in previous item), and Fong. Claim 111 has been cancelled because Applicant considers it to be redundant over existing rewritten claims.

51. The Rejection Of Claims 67-68 Under 35 USC § 103 On Dave (US 6,178,542 B1), Dave2 (US 6,086,628), And Nilsen (US 6,438,573 B1) Is Overcome

52. The last O.A. rejected dependent claim 96 on Dave, Dave2, and Nilsen. Claim 96 has been rewritten as new dependent claim 142 to define patentably over Dave, Dave2, and Nilsen and any combination thereof.
Applicant submits that claim 142 is independently patentable over Dave, Dave2, and Nilsen and any combination therefore for the same reasons as given in item 13 and item 15 above, and in addition, for the following reasons:

(1) There is no justification, in Dave, Dave2, and Nilsen, or in any other prior art separate from Applicant's disclosure, which suggests that these references be combined, much less combined in the manner proposed.
(2) Even if Dave, Dave2, And Nilsen were to be combined in the manner

proposed, the proposed combination would not show all of the novel physical features of claim 142.

(3) The proposed combination would not be physically possible or operative.

(4) The novel features of claim 142 produce new and unexpected results and hence are unobvious and patentable over these references.

52.1. Dave, Dave2, And Nilsen Do Not Contain Any Justification To Support Their Combination, Much Less In The Manner Proposed

With regard to the proposed combination of Dave, Dave2, And Nilsen, it is well known that in order for any prior-art references themselves to be validly combined for use in a prior-art § 103 rejection, *the references themselves* (or some other prior art) must suggest that they be combined. E.g., as was stated in In re Sernaker, 217 U.S.P.Q. 1, 6 (C.A.F.C. 1983):

> "[P]rior art references in combination do not make an invention obvious unless something in the prior art references would suggest the advantage to be derived from combining their teachings."

That the suggestion to combine the references should not come from applicant was forcefully stated in Orthopedic Equipment Co. v. United States, 217 U.S.PQ. 193, 199 (CAFC 1983):

> "It is wrong to use the patent in suit [here the patent application] as a guide through the maze of prior art references, combining the right references in the right way to achieve the result of the claims in suit [here the claims pending]. Monday morning quarterbacking is quite proper when resolving the question of nonobviousness in a court of law [here the PTO]."

As was further stated in Uniroyal, Inc. v. Rudkin-Wiley Corp., 5 U.S.P.Q.2d 1434 (C.A.F.C. 1988), "[w]here prior-art references require selective combination by the court

to render obvious a subsequent invention, there must be some reason for the combination other than the hindsight gleaned from the invention itself. . . . *Something in the prior art must suggest the desirability and thus the obviousness of making the combination.*" [Emphasis supplied.]

In line with these decisions, the Board stated in Ex Parte Levengood, 28 U.S.P.Q.2d 1300 [P.T.O.B.A.&]. 1993):

"In order to establish a *prima facie* case of obviousness, it is necessary for the examiner to present evidence, preferably in the form of some teaching, suggestion, incentive or inference in the applied prior art, or in the form of generally available knowledge, that one having ordinary skill in the art *would have been led to* combine the relevant teachings of the applied references in the proposed manner to arrive at the claimed invention. . . . That which is within the capabilities of one skilled in the art is not synonymous with obviousness, ... That one can *reconstruct* and/or explain the theoretical mechanism of an invention by means of logic and sound scientific reasoning does not afford the basis for an obviousness conclusion unless that logic and reasoning also supplies sufficient impetus to have led one of ordinary skill in the art to combine the teachings of the references to make the claimed invention. . . . Our reviewing courts have often advised the Patent and Trademark Office that it can satisfy the burden of establishing a *prima facie* case of obviousness only be showing some objective teaching in either the prior art, or knowledge generally available to one of ordinary skill in the art, that `would lead' that individual `to combine the relevant teachings of the references." ... Accordingly, an examiner cannot establish obviousness by locating references which describe various aspects of a patent application's invention without also providing evidence of the motivating force which would impel one skilled in the art to do what the patent applicant has done."

In the present case, there is no reason given in the last O.A. to support the proposed combination, other than the statement *"Referring to claim 96, Dave in view of Dave2 fails to explicitly teach a method as defined in claim 58, including restricting every*

periodic process in the pre-run-time schedule to be executed strictly within its time slot. However, Nilsen teaches executing within the allotted time slot and that exceeding it would be bad ("A difficulty arises, however, because the real-time executive desires to control exactly when a particular task is terminated in order to prevent one task's sin (i.e., exceeding its allotted time slot from corrupting the integrity of the entire system." col. 9, lines 5-15).It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of not exceeding the allotted time slot for the reason of increasing the integrity of the entire system ("A difficulty arises, however, because the real-time executive desires to control exactly when a particular task is terminated in order to prevent one task's sin (i.e., exceeding its allotted time slot from corrupting the integrity of the entire system." col. 9, lines 5-15).

However, the fact that the words "not exceeding the allotted time slot" *(Nilsen, col. 9, lines 5-1)* appear in the reference is not sufficient to gratuitously and selectively combine parts of one reference (Nilsen's real-time programming method) with another reference in order to meet Applicant's novel claimed combination of features.

Applicant therefore submits that combining Dave, Dave2, And Nilsen is not legally justified and is therefore improper. Thus Applicant submits that the rejection on these references is also improper and should be withdrawn.

Applicant respectfully requests, if the claims are again rejected upon any combination of references, that the Examiner include an explanation, in accordance with M.P.E.P. § 706.02, Ex parte Clapp, 27 U.S.P.Q. 972 (P.O.B.A. 1985), and Ex parte Levengood, supra, a "factual basis to support his conclusion that it would have been obvious" to make the combination,

52.2. Even if Dave, Dave2, And Nilsen Were To Be Combined In The Manner Proposed, The Proposed Combination Would Not Show All The Novel Features Of Claim 142.

However even if the combination of Dave, Dave2, And Nilsen were justified, claim 142 would still have novel and unobvious features over the proposed combination.

52.2.1. Neither Dave, nor Dave2, nor Nilsen, nor any possible combination thereof show the feature of restricting every periodic process in the pre-run-time schedule to be executed strictly within its time slot such that the predetermined constraints will be satisfied.

Claim 142 clearly distinguishes over Dave, Dave2, and Nilsen since claim 142 includes all the limitations of claim 115 and claim 115 recites:

"*scheduling on one or more processors, executions of a plurality of periodic and asynchronous processes, comprising:*

(A)

automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(1) worst-case computation times for periodic processes and asynchronous processes,

(2) period for periodic processes,

(3) minimum time between two consecutive requests for asynchronous processes,

(4) deadline for periodic processes and asynchronous processes,

(5) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and

ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(6) precedence relations for periodic processes wherein each precedence relation being defined between a pair of processes comprising a first process and a second process, both said first process and said second process being periodic processes, said first process precedes said second process, execution of said second process only allowed to start after said first process has completed its execution,

(7) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots, including the step of converting one or more asynchronous processes into corresponding new periodic processes prior to the mapping step, and mapping new periodic processes to time slots in a manner similar to mapping of other periodic processes, such that said predetermined constraints will be satisfied

(B)

during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to

schedule the process executions, such that said predetermined constraints will be satisfied."

and claim 142 further recites:

"*restricting every periodic process in the pre-run-time schedule to be executed strictly within its time slot.*"

Neither Dave, nor Dave2, nor Nilsen teach this.

52.2.1. As already explained in item 3, (1)-(8) in this amendment, both Dave and Dave2 are co-synthesis algorithms, and it is notoriously well known that co-synthesis is performed off-line, not during run-time when tasks are actually executed on a processor. Hence Dave and Dave do not teach scheduling the process executions during run-time, nor teach scheduling the process executions strictly within time slots during run-time.

52.2.2. Nilsen's teaching of not exceeding its allotted time slot is fundamentally different from "*restricting every periodic process in the pre-run-time schedule to be executed strictly within its time slot*" as defined in claim 142. Nilsen may in fact not allow a task to execute within its time slot at all by terminating, aborting, or suspending a task even before a task has any chance to execute in a time slot.
Nilsen may terminate, abort, or suspend a task in order to guarantee that a task does not exceed a time slot. *("A difficulty arises, however, because the real-time executive desires to control exactly when a particular* task is terminated *in order to prevent one task's sin (i.e., exceeding its allotted time slot from corrupting the integrity of the entire system." Nilsen, col. 9, lines 5-15). ("For a timed-statement control structure, it is anticipated that specified code segment will execute within the specified increment of time, execution otherwise being aborted." Nilsen, col. 3, lines 44-47.) ("Make sure there is sufficient time in the current time slice to execute the complete atomic statement. ... If this condition cannot be satisfied at the current time, suspend the task." Nilsen, col. 17, lines 53-54).*

Nilsen's method of not allowing processes to exceed time slots is completely foreign to Applicant's invention as defined in claim 142. In Applicant's invention as defined by claim 142, no process can ever be terminated, or aborted before it has completed its computation, otherwise the method will not be able to satisfy the predetermined constraints as defined by claim 142.

52.2.3. Claim 142 requires that the predetermined timing constraints, including deadlines, be satisfied, when "*restricting every periodic process in the pre-run-time schedule to be executed strictly within its time slot*". This is because claim 142 is dependent on claim 115, and claim 115 recites: "*...including satisfaction of predetermined constraints comprising*

(1) worst-case computation times for periodic processes and asynchronous processes,

(2) period for periodic processes,

(3) minimum time between two consecutive requests for asynchronous processes,

(4) deadline for periodic processes and asynchronous processes,"

Thus, in claim 142, none of the periodic processes can be terminated or aborted, otherwise they would not be able to satisfy the above predetermined constraints.

52.2.1. Neither Dave, nor Dave2, nor Nilsen, nor any possible combination thereof show the feature of automatically generating a pre-run-time schedule in which predetermined constraints comprising worst-case computation time, period, minimum time between two consecutive exclusion relations, deadline, permitted range of offset constraints, precedence relations are satisfied and during run-time executing the processes such that the predetermined constraints are satisfied. (Discussion on Dave's lack of ability to satisfy exclusion constraints is provided in item 11.1. above and is equally applicable to Dave and Dave2 and is hereby included here by reference.)

Claim 142 clearly distinguishes over Dave, Dave2, And Nilsen, or any possible combination thereof, since it is dependent on claim 115 which recites

"automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(1) worst-case computation times for periodic processes and asynchronous processes,

(2) period for periodic processes,

(3) minimum time between two consecutive requests for asynchronous processes,

(4) deadline for periodic processes and asynchronous processes,

(5) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(6) precedence relations for periodic processes wherein each precedence relation being defined between a pair of processes comprising a first process and a second process, both said first process and said second process being periodic processes, said first process precedes said second process, execution of said second process only allowed to start after said first process has completed its execution,

(7) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the

time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots, including the step of converting one or more asynchronous processes into corresponding new periodic processes prior to the mapping step, and mapping new periodic processes to time slots in a manner similar to mapping of other periodic processes, such that said predetermined constraints will be satisfied".

Neither Dave, nor Dave2, nor Nilsen, nor any possible combination thereof, show the feature of *automatically generating a pre-run-time schedule including satisfaction of predetermined constraints comprising*

"exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation".

52.2.1 (a) Nilsen does not enforce exclusion constraints. ("It does not enforce mutual exclusion." col. 27, line 5-6).

52.2.1 (b) Nilsen does not show "automatically generating a pre-run-time schedule".

52.2.1 (c) Nilsen does not show "automatically generating a pre-run-time schedule in which predetermined constraints comprising worst-case computation time, period, minimum time between two consecutive exclusion relations, deadline, permitted range of offset constraints, precedence relations are satisfied and during run-time executing the processes such that the predetermined constraints are satisfied."

52.2.1 (d) Neither Dave nor Dave2 satisfy exclusion constraints. (This has been shown for Dave in item 11.1. and is equally applicable to Dave and Dave2.)

52.2.2. Neither Dave, nor Dave2, nor Nilsen, nor any possible combination thereof show the feature of permitted range of offset constraints for periodic processes

Claim 142 which is dependent on claim 115 clearly distinguishes over any combination of Dave, Dave2, And Nilsen since claim 115 recites

"including satisfaction of predetermined constraints comprising

...

(4) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,"

Dave, nor Dave2, nor Nilsen, nor any possible combination thereof show the feature of satisfying predetermined constraints comprising "*permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,*"

52.3 The Suggested Combination Of Dave, Dave2, And Nilsen Would Not Be Physically Possible Or Operative.

There is plenty of unequivocal evidence that show that the suggested combination of Dave, Dave2, and Nilsen would not be physically possible or operative, here is just a few of them:

52.3.1. Nilsen does NOT provide any guarantee, either before run-time, nor during run-time that all specified timing constraints, including deadline constraints, will be satisfied:

Nilsen explicitly acknowledges that it is possible that the system's timing is not guaranteed:

(a1) The system may run behind schedule. *("Note that the real-time executive may need to revise the time budget dynamically (if, for example, the system finds itself running behind schedule.),"* Fig. 13, lines -5 to -4.)

(a2) Time alarms may be postponed indefinitely. *("If the alarm was set before entering into the current atomic segment and the alarm time arrives while the application is still executing the body of the atomic segment, delivery of the alarm's exception is postponed until after the application leaves its atomic statement." col. 18, lines 41-45.)*

(a3) Tasks can be terminated, aborted, and suspended at any time. *("A difficulty arises, however, because the real-time executive desires to control exactly when a particular* task is terminated *in order to prevent one task's sin (i.e., exceeding its allotted time slot from corrupting the integrity of the entire system." Nilsen, col. 9, lines 5-15). ("For a timed-statement control structure, it is anticipated that specified code segment will execute within the specified increment of time, execution otherwise being aborted." Nilsen, col. 3, lines 44-47.) ("Make sure there is sufficient time in the current time slice to execute the complete atomic statement. ... If this condition cannot be satisfied at the current time, suspend the task." Nilsen, col. 17, lines 53-54).*

(a4) Nilsen's method is not intended to result in a system such that all predetermined constraints including deadline constraints are satisfied

("The large majority of code comprising an RTPM program is not intended to be execution time analyzable." Nilsen, col. 6, lines 27-28.)

(a5) Some code segments of a task may never be assigned time to allow them to be executed. *("The RTPM also utilizes an atomic-statement control structure. ... the control structure requiring that either enough time be assigned to execute the specified code segment or that none be assigned." Nilsen, col. 3, lines 58-64.)*

52.3.2. The references of Dave and Dave2 do not contain any teaching, that suggests that their method could be physically combined with the teachings of Nilsen. Neither does the reference of Nilsen contain any teaching, that suggests the teaching of Nilsen could be physically combined with the teachings of Dave and Dave2.

52.3.3. There is no evidence in the prior art that suggests that the references of Dave, Dave2, And Nilsen could be combined to produce an operative method.

52.3.4. Combining Dave, Dave2, And Nilsen will NOT produce an operative method that will cover all the features of claim 142 as Nilsen does not guarantee that predetermined constraints including deadlines will be satisfied.

52.3.5. Even if Dave and Dave2, are operative when functioning alone, and even if it is structurally possible to combine them with Nilsen, the resulting method will be inoperative for the purpose of satisfying hard timing constraints at the very least because of the possibility of terminating, or aborting tasks before the tasks are completed, and the general inability to guarantee that timing constraints including deadline constraints will be satisfied in Nilsen's method.

The above evidence, clearly shows that the O.A.'s reason for combining Dave, Dave2, and Nilsen to find obviousness of claim 142, i.e., *("A difficulty arises, however, because*

the real-time executive desires to control exactly when a particular task is terminated *in order to prevent one task's sin (i.e., exceeding its allotted time slot from corrupting the integrity of the entire system." Nilsen, col. 9, lines 5-15).* is not a valid reason.

52.4. The Novel Features Of Claim 142 Produce New And Unexpected Results And Hence Are Unobvious And Patentable Over Any Possible Combination Of Dave, Dave 2, and Nilsen Under § 103

Claim 142 provides the following combination of features that have never before been provided simultaneously:

(1) Applicant's invention as defined by claim 142 provides the capability to enforce exclusion relations on pairs of processes that can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are mapped into time slots in the pre-run-time schedule, thus providing the capability to prevent errors caused by more than one process simultaneously accessing shared resources such as data in systems of one or more processors while also providing the capability to select precisely which pairs of process' executions should not overlap in time when they access shared data thus maximizing the system's flexibility in meeting deadline constraints.
Neither Dave, nor Dave 2, nor Nilsen show this feature.

(2) Applicant's invention as defined by claim 142 simultaneously provides the capability to increase the flexibility of meeting deadline constraints when there is flexibility in assigning an offset value for a periodic process.
Neither Dave, nor Dave 2, nor Nilsen show this feature.

(3) Applicant's invention as defined by claim 142 simultaneously provides the capability to increase the predictability of the system by restricting all periodic

processes in the pre-run-time schedule to execute strictly within its time slot while satisfying all the specified constraints for both periodic and asynchronous processes.
Neither Dave, nor Dave 2, nor Nilsen show this feature.

(3) Applicant's invention as defined by claim 142 provides the above combination of features, while providing the capability to handle processes that can be either (i) asynchronous processes that are not converted into new periodic processes and hence are not mapped into time slots in the pre-run-time schedule, or (ii) asynchronous processes that are converted into new periodic processes and hence are mapped into time slots in the pre-run-time schedule, or (iii) periodic processes that are mapped into time slots in the pre-run-time schedule, thus obtaining the advantages of pre-run-time scheduling including ability to use the information in the pre-run-time schedule to achieve greater predictability, ability to handle complex constraints, lower run-time overhead, and in general greatly increase the efficiency of scheduling; and the advantages of run-time scheduling including ability to handle asynchronous processes with very short deadlines that cannot be converted into periodic processes or will waste too much processor capacity if converted into periodic processes, while providing a guarantee that all the constraints, including hard deadlines, will be satisfied before run-time.
Neither Dave, nor Dave 2, nor Nilsen show this combination of features.

A. Applicant's invention achieves **unexpected results**: A system with all the above important features combined together, has never been realized before. The combination of results achieved by Applicant's invention are new and vastly superior compared to that of Dave, Dave2, and Nilsen or any combination thereof.

B. Applicant's invention is classified in a **crowded art** (a prior art patent cited by the O.A. states that, *"There is a vast amount of literature in the area of scheduling of soft and hard aperiodic tasks", Dave, col. 2, lines 47-48*); therefore, even a small step forward should be regarded as significant.

Applicant therefore submits that claim 142 is patentable under § 102 and § 103 and should be allowed, since they produce new and unexpected results over Dave, Dave2, and Nilsen or any combination thereof.

53. The Rejection Of Claim 71 Under 35 USC § 103 On Matsumoto (US5,448, 732) Is Overcome

54A The last O.A. rejected dependent claim 71 on Matsumoto. Claim 71 has been rewritten as new dependent claim 123 to define patentably over Matsumoto and any combination thereof. Applicant requests reconsideration of this rejection, as now applicable to claim 123, for the following reasons:

Claim 123 clearly distinguishes over Matsumoto since claim 123 recites:
"*determining whether an asynchronous process should or should not be converted into a new periodic process, comprising:*
(1) calculating a first processor capacity that is required for the asynchronous process if left unconverted,
(2) calculating a second processor capacity which is required to be reserved for the new periodic process,
(3) determining whether the ratio of said first processor capacity to said second processor capacity exceeds a predetermined threshold value."

Matsumoto does not do this.

Applicant respectfully disagrees with the last O.A.'s statement, "Referring to claim 71, Matsumoto teaches a method of determining whether each asynchronous process should or should not be converted into a new periodic process by calculating whether a ratio of processing capacity of the processor which is required to be reserved for new periodic processes, to processor capacity that is required for the asynchronous process if left

unconverted, exceeds a predetermined threshold value. *("Each of [1], [2], and [3] is a condition for improving the theoretical effectiveness, and each of [4] and [5] is a condition for doing the same by determining "n" heuristically, or from experience. Depending on the application which is running, "n" is adjusted in order to improve efficiency. With respect to conditions [4] and [5], instead of the number of processes waiting for synchronization, the ratio of the number of processors in the group to the number of processors waiting for synchronization in the group is used," col. 6, lines 25-35*). As mentioned earlier, it is common knowledge in the art of task management and process synchronization that converting asynchronous processes to synchronous ones is merely synchronization. However, "Official Notice" is taken that both the concept and advantages of providing that the use of thresholds is well known and expected in the art. It would have been obvious to one of ordinary skill in the art at the time the invention was made to include thresholds to the existing method for the reason of increasing the control by being able to set limits or boundaries which determine one state over another." (Last O.A., page 22, item 53-54.)

Applicant respectfully disagrees with the above "Official Notice", and requests that documentary proof be provided, and the data be stated as specifically as possible, and the facts be supported, under M.P.E.P Section 2144.03 and 37 CFR 1.104(d)(2) for the "Office Notice" position that the use of thresholds is well known and expected in the art in the context of claim 123, i.e.,

"*determining whether an asynchronous process should or should not be converted into a new periodic process, comprising:*

(1) calculating a first processor capacity that is required for the asynchronous process if left unconverted,

(2) calculating a second processor capacity which is required to be reserved for the new periodic process,

(3) determining whether the ratio of said first processor capacity to said second processor capacity exceeds a predetermined threshold value."

would have been obvious to one of ordinary skill in the art at the time the invention was made to include thresholds to the existing method for the reason of increasing the control

by being able to set limits or boundaries which determine one state over another. In this specific case, synchronization would begin after the threshold is reached."

(a) Matsumoto fails to show any of the important features of claim 123. The quoted phrases, "*Each of [1], [2], and [3] is a condition for improving the theoretical effectiveness, and each of [4] and [5] is a condition for doing the same by determining "n" heuristically, or from experience. Depending on the application which is running, "n" is adjusted in order to improve efficiency. With respect to conditions [4] and [5], instead of the number of processes waiting for synchronization, the ratio of the number of processors in the group to the number of processors waiting for synchronization in the group is used,"(Matsumoto col. 6, lines 25-35*) bear no relation to converting an asynchronous process to a new periodic process. It does not show how to calculate the processor capacity that is required for the asynchronous process if left unconverted, or the processor capacity which is required to be reserved for the new periodic process. Matsumoto fails to show any of the features of claim 123. Matsumoto does not have the notion of periodic processes, much less the notion of converting asynchronous processes into periodic processes.

(b) The art of how to convert an asynchronous process to a new periodic process is not merely "synchronization" as suggested by the last O.A., it is one of the most important, yet least understood, and under-studied techniques in the field of real-time computing. Applicant's papers related to real-time computing have been reprinted in two IEEE Computer Society Tutorial collections and are also widely referenced in textbooks on real-time systems. Applicant is internationally well-known as an expert in real-time computing, and Applicant has taken a special interest in this particular technique for over 20 years, yet it had taken Applicant many, many years before Applicant realized and invented the technique shown by claim 123, hence Applicant can attest to the fact that the technique shown in claim 123 is far, far from obvious.

(c) As can been seen in Fig. 26 of the drawings, and paragraph [0174], determining the ratio of processing capacity of the processor which is required to be reserved

for new periodic processes, to processor capacity that is required for the asynchronous process if left unconverted, requires an elaborate procedure that is far from obvious. Hence Applicant is not surprised at all by the fact that no prior art has been found that meets the features shown in claim 123, because, to Applicant's knowledge, up to even today, no one has published a similar invention.

(d) Matsumoto readily acknowledges that his method may include processes that are deadlocked *("All of the processes in the group of processes concerned are dispatched to processors and waiting for synchronization at one time; this event occurs due to programming errors (deadlock)."col. 6, lines 5-9.)* Thus any combination involving Matsumoto will be inoperative, not only because of deadlocks, but in general due to the inability to deal with any timing constraints.

54B Further Distinctions That Render Claim 155 Patentable Over Matsumoto (US5,448, 732)

The last O.A.item 54 rejected dependent claim 71 on Matsumoto. Claim 71 has also been rewritten as new dependent claim 155 in this amendment to define patentably over Matsumoto and any combination thereof. Applicant requests reconsideration of this rejection, as now applicable to claim 155, for the following reasons, in addition to all the reasons mentioned in item 54.1:

Claim 155 clearly distinguishes over Matsumoto since claim 155 recites:
"*A method of determining whether each asynchronous process in an original set of asynchronous processes should or should not be converted into a new periodic process and included in a set of periodic processes that will be mapped to time slots in a pre-run-time schedule,*

each asynchronous process in said original set of asynchronous processes has predetermined asynchronous process constraints comprising worst-case computation time, deadline, and minimum time between two consecutive requests constraints,

each periodic process in said set of periodic processes has predetermined periodic process constraints comprising permitted range of offset, worst-case computation time, deadline, period constraints,

each new periodic process has new periodic process constraints comprising permitted range of offset, worst-case computation time, deadline, period constraints,

said method steps comprising:

(A)

selecting one asynchronous process in said original set of asynchronous processes,

(1) tentatively converting said one asynchronous process into a corresponding new periodic process, such that said asynchronous process constraints of said one asynchronous process will be satisfied by said new periodic process constraints of said corresponding new periodic process,

(2) calculating a first processor capacity which is required to be reserved if said one asynchronous process is not converted,

(3) calculating a second processor capacity which is required to be reserved if said one asynchronous process is converted,

(4) including a copy of said one asynchronous process in a second set of asynchronous processes if the ratio of said first processor capacity to said second processor capacity exceeds a predetermined threshold, otherwise including said tentatively converted corresponding new periodic process in said set of periodic processes,

(B)

repeating A until every asynchronous processes in said original set of asynchronous processes has been selected,

(C)

repeating B,

including removing from the periodic set any corresponding new periodic process that was included in the periodic set in any previous execution of B whenever a copy of a asynchronous process is to be included in the second set of asynchronous processes in the current execution of step B,

further including removing any copy of an asynchronous process from the second asynchronous set that was included in the second asynchronous set in any previous execution of step A whenever any corresponding new periodic process is to be included in the set of periodic processes in the current execution of step A,

(D)

terminating when no more changes to the second asynchronous set have occurred between the beginning and the end of any execution of C."

Matsumoto does not do this.

Applicant respectfully disagrees with the last O.A.'s statement, "Referring to claim 71, Matsumoto teaches a method of determining whether each asynchronous process should or should not be converted into a new periodic process by calculating whether a ratio of processing capacity of the processor which is required to be reserved for new periodic processes, to processor capacity that is required for the asynchronous process if left unconverted, exceeds a predetermined threshold value. *("Each of [1], [2], and [3] is a condition for improving the theoretical effectiveness, and each of [4] and [5] is a condition for doing the same by determining "n" heuristically, or from experience. Depending on the application which is running, "n" is adjusted in order to improve efficiency. With respect to conditions [4] and [5], instead of the number of processes waiting for synchronization, the ratio of the number of processors in the group to the number of processors waiting for synchronization in the group is used," col. 6, lines 25-35*). As mentioned earlier, it is common knowledge in the art of task management and process synchronization that converting asynchronous processes to synchronous ones is merely synchronization. However, "Official Notice" is taken that both the concept and advantages of providing that the use of thresholds is well known and expected in the art. It would have been obvious to one of ordinary skill in the art at the time the invention was made to include thresholds to the existing method for the reason of increasing the control by being able to set limits or boundaries which determine one state over another." (Last O.A., page 22, item 53-54.)

Applicant respectfully disagrees with the above "Official Notice", and requests that documentary proof be provided, and the data be stated as specifically as possible, and the facts be supported, under M.P.E.P Section 2144.03 and 37 CFR 1.104(d)(2) for the "Office Notice" position that the use of thresholds is well known and expected in the art in the context of claim 155, i.e.,

"A method of determining whether each asynchronous process in an original set of asynchronous processes should or should not be converted into a new periodic process and included in a set of periodic processes that will be mapped to time slots in a pre-run-time schedule,

each asynchronous process in said original set of asynchronous processes has predetermined asynchronous process constraints comprising worst-case computation time, deadline, and minimum time between two consecutive requests constraints,

each periodic process in said set of periodic processes has predetermined periodic process constraints comprising permitted range of offset, worst-case computation time, deadline, period constraints,

each new periodic process has new periodic process constraints comprising permitted range of offset, worst-case computation time, deadline, period constraints,

said method steps comprising:

(A)

selecting one asynchronous process in said original set of asynchronous processes,

(1) tentatively converting said one asynchronous process into a corresponding new periodic process, such that said asynchronous process constraints of said one asynchronous process will be satisfied by said new periodic process constraints of said corresponding new periodic process,

(2) calculating a first processor capacity which is required to be reserved if said one asynchronous process is not converted,

(3) calculating a second processor capacity which is required to be reserved if said one asynchronous process is converted,

(4) including a copy of said one asynchronous process in a second set of asynchronous processes if the ratio of said first processor capacity to said second processor capacity exceeds a predetermined threshold, otherwise including said

tentatively converted corresponding new periodic process in said set of periodic processes,

(B)

repeating A until every asynchronous processes in said original set of asynchronous processes has been selected,

(C)

repeating B,
including removing from the periodic set any corresponding new periodic process that was included in the periodic set in any previous execution of B whenever a copy of a asynchronous process is to be included in the second set of asynchronous processes in the current execution of step B,
further including removing any copy of an asynchronous process from the second asynchronous set that was included in the second asynchronous set in any previous execution of step A whenever any corresponding new periodic process is to be included in the set of periodic processes in the current execution of step A,

(D)

terminating when no more changes to the second asynchronous set have occurred between the beginning and the end of any execution of C."

(a) Matsumoto fails to show any of the important features of claim 155. The quoted phrases, "*Each of [1], [2], and [3] is a condition for improving the theoretical effectiveness, and each of [4] and [5] is a condition for doing the same by determining "n" heuristically, or from experience. Depending on the application which is running, "n" is adjusted in order to improve efficiency. With respect to conditions [4] and [5], instead of the number of processes waiting for synchronization, the ratio of the number of processors in the group to the number of processors waiting for synchronization in the group is used,"(Matsumoto col. 6, lines 25-35*) bear no relation to converting an asynchronous process to a new periodic process. It does not show how to calculate the processor capacity that is required for the asynchronous process if left unconverted, or the processor

capacity which is required to be reserved for the new periodic process. Matsumoto fails to show any of the features of claim 155. Matsumoto does not have the notion of periodic processes, much less the notion of converting asynchronous processes into periodic processes while satisfying the timing constraints. Matsumoto does not even have the notion of timing constraints, such as periods, worst-case computation times, deadlines, etc.

(b) The art of how to convert an asynchronous process to a new periodic process is not merely "synchronization" as suggested by the last O.A., it is one of the most important, yet least understood, and under-studied techniques in the field of real-time computing. Applicant's papers related to real-time computing have been reprinted in two IEEE Computer Society Tutorial collections and are also widely referenced in textbooks on real-time systems. Applicant is internationally well-known as an expert in real-time computing, and Applicant has taken a special interest in this particular technique for over 20 years, yet it had taken Applicant many years before Applicant realized and invented the technique shown by claim 155, hence Applicant can attest to the fact that the technique shown in claim 155 is far from obvious.

(c) As can been seen in Fig. 26 of the drawings, and paragraph [0174], determining the ratio of processing capacity of the processor which is required to be reserved for new periodic processes, to processor capacity that is required for the asynchronous process if left unconverted, requires an elaborate procedure that is far from obvious. Hence Applicant is not surprised at all by the fact that no prior art has been found that meets the features shown in claim 155, because, to Applicant's knowledge, up to even today, no one has published a similar invention.

(d) Matsumoto readily acknowledges that his method may include processes that are deadlocked *("All of the processes in the group of processes concerned are dispatched to processors and waiting for synchronization at one time; this event occurs due to programming errors (deadlock)."col. 6, lines 5-9.)* Thus any combination involving Matsumoto will be inoperative, not only because of deadlocks, but in general due to the inability to deal with any timing constraints.

55. **The Rejection Of Claim 108 Under 35 USC § 103(a) On Dave (US 6,178,542 B1), Dave2 (US 6,086,628), Lindsley (US 6,430,593 B1), and Matsumoto (US5,448, 732) Is Overcome**

56. The last O.A. item 55-56 rejected dependent claim 108 on Dave, Dave2, Lindsley, and Matsumoto. Claim 108 has also been rewritten as new dependent claim 156 to define patentably over Dave, Dave2, Lindsley, and Matsumoto and any combination thereof.

Applicant requests reconsideration of this rejection, as now applicable to claim 156, for the same reasons as stated in item 13, item 15, item 44 of this amendment, and further because Dave2 does not teach anything different from Dave that is of significance related to the patentability of claim 156.

Claim 156 clearly distinguishes over Dave, Dave2, Lindsley, and Matsumoto since claim 156 includes all the limitations of claim 124, and claim 124 recites:

"*prior to generating the pre-run-time schedule, determining whether each asynchronous process should or should not be converted into a new periodic process, converting a subset of a predetermined set of asynchronous processes having worst-case computation time, deadline, minimum time between two requests constraints, which have been determined to be convertible, into a set of new periodic processes having worst-case computation time, period, deadline, permitted range of offset constraints and reducing possible timing conflicts with other periodic or asynchronous processes with less latitude in meeting their deadlines, by taking into consideration the computation time requirements of the latter processes when determining the deadline of the new periodic process.*"

and claim 156 further recites:

"*A method as defined in claim 124, in which the determining step is performed by (1) calculating a first processing capacity of the processor which is required to be reserved for the new periodic process,*

(2) calculating a second processor capacity that is required for the asynchronous process if left unconverted,

(3) determining whether the ratio of said first processing capacity to said second processing capacity exceeds a predetermined threshold value."

As explained in item 44 of this amendment, neither Dave, nor Dave2, nor Lindsley, nor Matsumoto do this.

(a) Matsumoto fails to show any of the important features of claim 156. The quoted phrases, "*of processes waiting for synchronization, the ratio of the number of processors in the group to the number of processors waiting for synchronization in the group is used,"(* Matsumoto *col. 6, lines 25-35*) bear no relation to converting an asynchronous process to a new periodic process. It does not show how to calculate the processor capacity that is required for the asynchronous process if left unconverted, or the processor capacity which is required to be reserved for the new periodic process. Matsumoto fails to show any of the features of claims 115 and 124, on which 156 is based. Matsumoto does not even have the notion of timing constraints, such as periods, worst-case computation times, deadlines, etc.

(b) The art of how to convert an asynchronous process to a new periodic process is not merely "synchronization" as suggested by the last O.A., it is one of the most important, yet least understood, and under-studied techniques in the field of real-time computing. Applicant's papers related to real-time computing have been reprinted in two IEEE Computer Society Tutorial collections and are also widely referenced in textbooks on real-time systems. Applicant is internationally well-known as an expert in real-time computing, and Applicant has taken a special interest in this particular technique for over 20 years, yet it had taken Applicant many years before Applicant realized and invented the technique shown by claim 156, hence Applicant can attest to the fact that the technique shown in claim 156 is far from obvious.

(c) As can been seen in Fig. 26 of the drawings, and paragraph [0174], determining the ratio of processing capacity of the processor which is required to be reserved

for new periodic processes, to processor capacity that is required for the asynchronous process if left unconverted, requires an elaborate procedure that is far from obvious. Hence Applicant is not surprised at all by the fact that no prior art has been found that meets the features shown in claim 156, because, to Applicant's knowledge, up to even today, no one has published a similar invention.

(d) Matsumoto readily acknowledges that his method may include processes that are deadlocked *("All of the processes in the group of processes concerned are dispatched to processors and waiting for synchronization at one time; this event occurs due to programming errors (deadlock)."col. 6, lines 5-9.)* Thus any combination involving Matsumoto will be inoperative, not only because of deadlocks, but in general due to the inability to deal with any timing constraints.

57. The Rejection Of Claim 114 Under 35 USC § 103(a) On Dave (US 6,178,542 B1), Lindsley (US 6,430,593 B1), and Matsumoto (US5,448, 732) Is Overcome

58. The last O.A. rejected dependent claim 114 on Dave, Lindsley, and Matsumoto. Claim 114 has been rewritten as new dependent claim 152 to define patentably over Dave, Lindsley, and Matsumoto and any combination thereof.
Applicant requests reconsideration of this rejection, as now applicable to claim 152, for the following reasons, in addition to to the reasons given in item11, item 11.2.1. and 11.2.2, and further in addition to all the reasons mentioned in item 11, item 13, item 15 in this amendment:

Claim 152 clearly distinguishes over Dave, Lindsley, and Matsumoto since claim 152 includes all the limitations of claim 151, and claim 151 recites:
"*A method as defined in claim 150 including, prior to the mapping step, converting one or more asynchronous processes having a worst-case computation time, deadline, minimum time between two requests constraints, into a set of new periodic processes, and reducing possible timing conflicts with other periodic or asynchronous processes with less latitude in meeting their deadlines, by taking into consideration the computation time*

requirements of the latter processes when determining the deadline of each of the new periodic processes, and mapping the new periodic process in a manner similar to mapping of other periodic processes."

and claim 152 further recites:

"*A method as defined in claim 151, in which the determining step is performed by calculating whether a ratio of processing capacity of the processor which is required to be reserved for new periodic processes, to processor capacity that is required for the asynchronous process if left unconverted, exceeds a predetermined threshold value.*"

Neither Dave, nor Lindsley, nor Matsumoto do this.

(a) Matsumoto fails to show any of the important features of claim 152. The quoted phrases, "*of processes waiting for synchronization, the ratio of the number of processors in the group to the number of processors waiting for synchronization in the group is used,"(* Matsumoto *col. 6, lines 25-35*) bear no relation to converting an asynchronous process to a new periodic process. It does not show how to calculate the processor capacity that is required for the asynchronous process if left unconverted, or the processor capacity which is required to be reserved for the new periodic process. Matsumoto fails to show any of the features of claims 115 and 124, on which 152 is based. Matsumoto does not even have the notion of timing constraints, such as periods, worst-case computation times, deadlines, etc.

(b) The art of how to convert an asynchronous process to a new periodic process is not merely "synchronization" as suggested by the last O.A., it is one of the most important, yet least understood, and under-studied techniques in the field of real-time computing. Applicant's papers related to real-time computing have been reprinted in two IEEE Computer Society Tutorial collections and are also widely referenced in textbooks on real-time systems. Applicant is internationally well-known as an expert in real-time computing, and Applicant has taken a special interest in this particular technique for over 20 years, yet it had taken Applicant many, many years before Applicant realized and invented the technique shown by

claim 152, hence Applicant can attest to the fact that the technique shown in claim 152 is far, far from obvious.

(c) As can been seen in Fig. 26 of the drawings, and paragraph [0174], determining the ratio of processing capacity of the processor which is required to be reserved for new periodic processes, to processor capacity that is required for the asynchronous process if left unconverted, requires an elaborate procedure that is far from obvious. Hence Applicant is not surprised at all by the fact that no prior art has been found that meets the features shown in claim 152, because, to Applicant's knowledge, up to even today, no one has published a similar invention.

(d) Matsumoto readily acknowledges that his method may include processes that are deadlocked *("All of the processes in the group of processes concerned are dispatched to processors and waiting for synchronization at one time; this event occurs due to programming errors (deadlock)."col. 6, lines 5-9.)* Thus any combination involving Matsumoto will be inoperative, not only because of deadlocks, but in general due to the inability to deal with any timing constraints.

59. Claim 153 Overcomes Rejection Of Claim 58 Under 35 USC § 103 On Dave (US 6,178,542 B1) and Dave2 (US 6,086,628)

The last O.A. item 12-13 rejected independent claim 58 on Dave and Dave2. Claim 58 has also been rewritten as new independent claim 153 (in addition to independent claim 116), to define patentably over Dave and Dave2 and any combination thereof. Applicant requests reconsideration of this rejection, as now applicable to claim 153, for the following reasons:

59.1. Neither Dave nor Dave2 show the feature in claim 153 of scheduling on one or more processors, executions of a plurality of periodic processes comprising during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such

that the predetermined constraints will be satisfied, where the predetermined constraints include permitted range of offset constraints and exclusion relations.

Claim 153 clearly distinguishes over Dave and Dave2 since it recites:

"scheduling on one or more processors, executions of a plurality of periodic processes, comprising:

(A)

automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(1) worst-case computation times,

(2) period,

(3) deadline,

(4) permitted range of offset constraints wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(5) precedence relations wherein each precedence relation being defined between a pair of processes comprising a first process and a second process, execution of said second process only allowed to start after said first process has completed its execution,

(6) exclusion relations wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process

excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots,

(B)

during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions such that said predetermined constraints will be satisfied."

Neither Dave nor Dave2 show any feature related to *"scheduling on one or more processors, executions of a plurality of periodic processes, comprising during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such that said predetermined constraints will be satisfied", "said predetermined constraints comprising: ... permitted range of offset constraints ..., exclusion relations... ."*

The last O.A. cited from Dave: *("scheduling", "tasks", "execution", "start", "finish", col. 9, lines 65-67)* as evidence that Dave shows the feature "(B) during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such that said predetermined constraints will be satisfied." Applicant submits that this is a misunderstood reference, as the reference clearly does not teach what the O.A. relies upon it as supposedly teaching.

Note that the phrases cited by the last O.A. are within the context of Section 4 The CASPER Algorithm in Dave: *("4 The CASPER Algorithm This section provides an overview of CASPER. FIG. 4 presents one possible co-synthesis process flow for the present invention. This flow is divided up into two-parts: pre-processing and synthesis. ... The synthesis step determines the allocation for both periodic and aperiodic graphs.*

The synthesis has two loops: 1) an outer loop for allocating each cluster, and an inner loop for evaluating various allocations for each cluster. For each cluster, an allocation array consisting of the possible allocations at that step is created. ... The next step is scheduling which determines the relative ordering of tasks/edges for execution and the start time and finish times for each task and edge. The algorithm employs a combination of both preemptive and non-preemptive static scheduling. Preemptive scheduling is used in restricted scenarios to minimize scheduling complexity (See Section 4.4.) For task preemption, the algorithm takes into consideration the operating system overheads such as interrupt overhead, context switch, remote procedure call (RPC) etc. through a parameter called preemption overhead (this information is experimentally determined and provided a priori). Incorporating scheduling into the inner loop facilitates accurate performance evaluation. Performance evaluation of an allocation is extremely important in picking the best allocation. An important step of performance evaluation is finish-time estimation. In this step, with the help of the scheduler, the finish times of each task and edge are estimated using the longest path algorithm. See Reference (2). After finish-time estimation, it is verified whether the given deadlines in the task graphs are met. The allocation evaluation step compares the current allocation against previous ones based on total dollar cost of the architecture." Dave, col. 9, lines 32-67, to col. 10, lines 1- 20).

Note that:

(1) All the phrases cited by the last O.A. *("scheduling", "tasks", "execution", "start", "finish", col. 9, lines 65-67)* are part of the Section 4 of Dave above that describe the "CASPER *(Co-synthesis of Aperiodic SPecification of Embedded system aRchitectures, Dave, col. 3, lines 56-57.)"* algorithm, which as its name describes, is a co-synthesis algorithm.

(2) Co-synthesis is a procedure of partitioning a system specification into hardware and software modules, which is notoriously well-known as a procedure that is done off-line, and NOT a procedure that actually executes tasks on a processor at run-time. *("Hardware-software co-synthesis is the process of partitioning an embedded system specification into hardware and software modules to meet performance, power, and cost goals." Dave, Abstract, lines 1- 4.)*

(3) The activity of "scheduling" *("scheduling", "tasks", "execution", "start", "finish", col. 9, lines 65-67)* cited by the last O.A. is part of the "inner loop" of the CASPER co-synthesis algorithm (*"Incorporating scheduling into the inner loop facilitates accurate performance evaluation" Dave, col. 10, lines 10-11*) and is only used to "evaluate" various allocations for each cluster *("The synthesis has two loops: 1) an outer loop for allocating each cluster, and an inner loop for evaluating various allocations for each cluster". Dave, col. 9, lines 55-58).*, so it is also part of co-synthesis that is done off-line, and it is NOT an activity that actually executes tasks on a processor at run-time.

(4) Fig. 4 (Sheet 3 of Dave), which shows CASPER, also shows that the activities associated with the phrases cited by the last O.A. *("scheduling", "tasks", "execution", "start", "finish", col. 9, lines 65-67),* which are part of "PERFORM SCHEDULING AND FINISH-TIME ESTIMATION" in the " inner loop" of Fig. 4, also clearly indicate that these activities are only part of an off-line algorithm to partition a system specification into a "FINAL SOLUTION", and do NOT actually executes tasks on a processor at run-time.

Similar to Dave, Dave2 is also a co-synthesis algorithm that is performed off-line and does not actually execute tasks on a processor at run-time:

Note that:

(5) All the phrases cited by the last O.A. *("scheduling", "tasks", "execution", "start", "finish", Dave, col. 9, lines 65-67)* are also part of the Section 2 of Dave2 that describe the COSYN algorithm *(The present invention is related to a heuristic-based co-synthesis technique, called COSYN, which includes allocation, scheduling, and performance estimation steps as well as power optimization features, Dave2, col. 2, lines 49-52.)" ("The next step is scheduling which determines the relative ordering of tasks/edges for execution and the start time and finish times for each task and edge."Dave2, col. 8, lines 56-58.)*, which as its name describes, is a co-synthesis algorithm.

(6) Co-synthesis is a procedure of partitioning a system specification into hardware and software modules, which is notoriously well-known as a procedure that is done off-line, and NOT a procedure that actually executes tasks on a processor at run-time. *("Hardware-software co-synthesis is the process of partitioning an embedded system specification into hardware and software modules to meet performance, power, and cost goals." Dave2, Abstract, lines 1- 4.)*

(7) The activity of "scheduling" *("scheduling", "tasks", "execution", "start", "finish", Dave, col. 9, lines 65-67)* cited by the last O.A. is also part of the "inner loop" of the COSYN co-synthesis algorithm of Dave2 (*"Incorporating scheduling into the inner loop facilitates accurate performance evaluation" Dave2, col. 8, lines 64-66*) and is only used to "evaluate" various allocations for each cluster *("The synthesis has two loops: 1) an outer loop for allocating each cluster, and an inner loop for evaluating various allocations for each cluster". Dave2, col. 8, lines 41-43).*, so it is also part of co-synthesis that is done off-line, and it is NOT an activity that actually executes tasks on a processor at run-time.

(8) Fig. 3 (Sheet 3 of Dave2), which shows COSYN *("2 The COSYN Algorithm In this section, an overview of the COSYN algorithm is provided followed up by details on each ste. FIG. 3 presents a co-synthesis process flow, according to one embodiment of the present invention." Dave2, col. 8, lines 14-18.)* also shows that the activities associated with the phrases cited by the last O.A. *("scheduling", "tasks", "execution", "start", "finish", Dave, col. 9, lines 65-67), ("The next step is scheduling which determines the relative ordering of tasks/edges for execution and the start time and finish times for each task and edge."Dave2, col. 8, lines 56-58.)* which is clearly shown as the box labeled "SCHEDULING" in the " inner loop" of Fig. 3, also clearly indicate that these activities are only part of an off-line algorithm to partition a system specification into a "FINAL SOLUTION" (Fig. 3), and do NOT actually executes tasks on a processor at run-time.

The above clearly and unequivocally shows that the references Dave and Dave2 cited by the O.A. do NOT show the feature in claim 153 of "(B) during run-time using the

information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such that said predetermined constraints will be satisfied."

59.1.(a) Claim 153 recites during run-time using the information in the pre-run-time schedule to schedule the process executions, which include periodic processes. Neither Dave nor Dave2 show during run-time scheduling periodic processes. In Dave and Dave 2, both synthesis and scheduling is performed off-line as explained in (1)-(4) above.

59.1.(b) Claim 153 recites during run-time using the information in the pre-run-time schedule to schedule the process executions, which include periodic processes, such that said predetermined constraints will be satisfied.
Dave does not show during run-time using the information in the pre-run-time schedule to schedule the process executions, which include periodic processes such that said predetermined constraints will be satisfied. In Dave and Dave2, both synthesis and scheduling is performed off-line, so no constraints are considered at run-time.

59.1.(c) Claim 153 recites during run-time scheduling of periodic processes, such that said predetermined constraints will be satisfied, where the predetermined constraints include permitted range of offset constraints and exclusion relations.
Neither Dave nor Dave2 show during run-time scheduling of periodic processes such that said predetermined constraints will be satisfied where the predetermined constraints include permitted range of offset constraints and exclusion relations. Neither Dave nor Dave2 consider permitted range of offset constraints and exclusion relations as defined in claim 153. This has been explained earlier in this amendment.

59.2. Neither Dave nor Dave2 show the feature of satisfying constraints comprising

exclusion relations and therefore does not provide the capability to prevent errors caused by more than one periodic process simultaneously accessing shared resources such as data while maximizing the system's flexibility in meeting deadline constraints.

(Item 11.1. shows Dave's lack of this feature and is equally applicable to Dave and Dave2, and is hereby included here by reference.)

Claim 153 clearly distinguishes over Dave and Dave2 since it recites

"including satisfaction of predetermined constraints comprising

...

(5) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,"

Neither Dave nor Dave2 show the feature of satisfying predetermined constraints comprising "*exclusion relations wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,"* and therefore does not provide the capability to prevent errors caused by more than one process simultaneously accessing shared resources such as data while maximizing the system's flexibility in meeting deadline constraints, while the processes are periodic processes that are mapped into time slots in the pre-run-time schedule.

The last O.A. citation of the features of Dave *("mapping of tasks to processing elements", "finish time", "constraints", col. 1, lines 50-67, and "co-simulation", col. 2, lines 17-43, and "periodic task graphs", "deadlines", col. 4, lines 53-67, and "finish-time estimation step is enhanced by employing a deadline-based scheduling technique", col. 5, lines 1-7, "worst-case execution times", "mapping tasks", col. 5, lines 25-46, "start time", "period", "deadline", col. 5, lines 53-67, "execution time slots", col. 7, lines 40-54),* does not show the Applicant's invention feature of exclusion constraints as cited above.

(Discussion on Dave's lack of this feature is provided in item 11.1. and is equally applicable to Dave and Dave2, and is hereby included here by reference.)

59.3. Neither Dave nor Dave2 show the feature of permitted range of offset constraints for periodic processes

Claim 153 clearly distinguishes over Dave and Dave2 since it recites

"including satisfaction of predetermined constraints comprising

...

(4) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value, n,"

Neither Dave nor Dave2 show the feature of satisfying predetermined constraints comprising "*permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower*

bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value, n,"

The last O.A. cited the following features of Dave2 *("An association array has an entry for each task of each copy of the task graph and contains information such as: 1) the PE to which it is allocated, 2) its priority level, 3) its deadline, 4) its best-case projected finish time (PFT), and its 5) its worst-case PFT. The deadline of the nth instance of a task is offset by (n-1) multiplied by its period from the deadline in the original task. The association array not only eliminates the need to replicate the task graphs, but it also allows allocation of different task graphs copies to different PEs, if desirable to derive en efficient architecture. This array also supports pipelining of task graphs, which is explained later in this specification.", col. 10, lines 1-12*). The last O.A. said that the above reference teaches using any offset value in a permitted range of offsets. Applicant submits that the above reference is a misunderstood reference, as the reference does not teach what the O.A. relies upon it as supposedly teaching.

Note that:

59.3.1. The term "offset" in the reference above refers to the duration of the time interval between the deadline of the nth instance in the nth period of the same task and the deadline of the first instance in the first period of the task.

59.3.2. In contrast, in Applicant's invention as defined by claim 153, the term "offset" refers to *"the duration of the time interval between the beginning of the first period of said periodic process and time zero" (Constraint (4) of Applicant's claim 153).*

35.2.3. The meaning of the term "offset" in Dave2 is clearly completely different from the meaning of the term "offset" in Applicant's invention as defined by claim 153. Thus the use of the term "offset" in Dave2 does not provide evidence that Dave2 shows the feature of "permitted range of offsets for periodic processes" in claim 153.

59.4. The Novel Features Of Claim 153 Produce New And Unexpected Results And Hence Are Unobvious And Patentable Over Dave and Dave 2 Under § 103

Claim 153 provides the following combination of features that have never before been provided simultaneously:

(1) Applicant's invention as defined by claim 153 provides the capability to enforce exclusion relations on pairs of processes thus providing the capability to prevent errors caused by more than one process simultaneously accessing shared resources such as data in systems of one or more processors while also providing the capability to select precisely which pairs of process' executions should not overlap in time when they access shared data thus maximizing the system's flexibility in meeting deadline constraints.
Neither Dave nor Dave2 show this feature.

(2) Applicant's invention as defined by claim 153 simultaneously provides the capability to increase the flexibility of meeting deadline constraints when there is flexibility in assigning an offset value for a periodic process and a permitted range of set constraint has been specified for each periodic process.
Neither Dave nor Dave2 show this feature.

(3) Applicant's invention as defined by claim 153 provides the above combination of features, while providing the capability, both before run-time, and during run-time, to schedule processes that are periodic processes that are mapped into time slots in the pre-run-time schedule, thus obtaining the advantages of pre-run-time scheduling including ability to use the information in the pre-run-time schedule to achieve greater predictability, ability to handle complex constraints, lower run-time overhead, and in general greatly increase the efficiency of scheduling; while providing a guarantee that all the constraints, including worst-case computation time, period, deadline, permitted range of offset, precedence relations and exclusion releations, will be satisfied before run-time.
Neither Dave nor Dave2 do not show this combination of features.

A. Applicant's invention achieves **unexpected results**: A system with all the above important features combined together, has never been realized before. The combination of results achieved by Applicant's invention are new and vastly superior compared to that of Dave and Dave2, or any combination thereof.

B. Applicant's invention is classified in a **crowded art** (a prior art patent cited by the O.A. states that, *"There is a vast amount of literature in the area of scheduling of soft and hard aperiodic tasks", Dave, col. 2, lines 47-48*); therefore, even a small step forward should be regarded as significant.

Applicant therefore submits that claim 153 is patentable under § 102 and § 103 and should be allowed, since they produce new and unexpected results over Dave and Dave2, or any combination thereof.

60. Claim 154 Overcomes Rejection Of Claim 127 Under 35 USC § 103 On Dave (US 6,178,542 B1), Dave2 (US 6,086,628) And Lindsley (US 6,430,593 B1)

The last O.A. item 24 rejected dependent claim 78 on Dave, Dave2, and Lindsley. Claim 78 has also been rewritten as new dependent claim 154 (in addition to dependent claim 127), to define patentably over Dave, Dave2, and Lindsley and any combination thereof. Applicant requests reconsideration of this rejection, as now applicable to claim 154, for the following reasons, in addition to the reasons given in item 24, and further in addition to the reasons given in item 35 and item 12 of this amendment.

Claim 154 clearly distinguishes over Dave, Dave2, and Lindsley since it recites:

"A method as defined in claim 153, including generating the pre-run-time schedule as a feasible two-part pre-run-time-schedule for execution of periodic processes that may have non-zero offsets (a) an initial part which may be of zero length, and (b) a repeating part having length which is equal to a least common multiple of lengths of the periods of the periodic processes,

all executions of all periodic processes within a time interval of length equal to the length of the least common multiple of the periodic process periods being included in the repeating part of the pre-run-time schedule, wherein all said predetermined constraints being satisfied for all executions of all periodic processes within said initial part and said repeating part."

Neither Dave nor Dave2 nor Lindsley do this, as explained by the reasons given in item 24, and further by the reasons given in item 59 and item 12 of this amendment.

61. Claim 157 Overcomes Rejection Of Claim 127 Under 35 USC § 103 On Dave (US 6,178,542 B1), Dave2 (US 6,086,628), Lindsley (US 6,430,593)*, and Fong (US 6,345,287 B1)

The last O.A. item 49 rejected dependent claim 80 on Dave, Dave2, Lindsley, and Fong. Claim 80 has been rewritten as new dependent claim 157 (in addition to claim 130) to define patentably over Dave, Dave2, Lindsley, and Fong and any combination thereof. Applicant requests reconsideration of this rejection, as now applicable to claim 157, for the following reasons, in addition to the reasons given in item 49:

Claim 157 clearly distinguishes over Dave, Dave2, Lindsley, and Fong since claim 157 includes all the limitations of claim 154, and claim 154 recites:
"A method as defined in claim 153, including generating the pre-run-time schedule as a feasible two-part pre-run-time-schedule for execution of periodic processes that may have non-zero offsets (a) an initial part which may be of zero length, and (b) a repeating part having length which is equal to a least common multiple of lengths of the periods of the periodic processes,
all executions of all periodic processes within a time interval of length equal to the length of the least common multiple of the periodic process periods being included in the repeating part of the pre-run-time schedule, wherein all said predetermined constraints being satisfied for all executions of all periodic processes within said initial part and said repeating part."

Claim 157 further recites:

"*A method as defined in claim 154, further including the steps of*

(A)

constructing a first schedule for executions of the periodic processes within an interval starting from zero and having length equal to maximum offset value plus a bounded number of times of the length of a least common multiple of the periodic process periods, conditions for determining feasibility requiring the existence of a point in said first schedule wherein starting from the latter point the schedule repeats in subschedule interval lengths equal to a least common multiple of lengths of the periodic process periods, timing of all executions of all periodic processes within a time interval having length equal to the length of the least common multiple of the periodic process periods being included in each said repeating subschedule interval, and including satisfaction of all predetermined constraints for all executions of all periodic processes within the subschedule interval starting from time zero and ending at said point plus the length of the least common multiple of the periodic process periods in said first schedule, and checking for the first occurrence of said point in said first schedule,

(B)

generating said feasible two-part pre-run-time-schedule by

(1) using a subschedule interval starting from time zero and ending at said point in said first schedule as said initial part of said feasible two-part pre-run-time schedule, and

(2) using a subschedule interval starting from said point and ending at said point plus the length of the least common multiple of the periodic process periods in said first schedule as said repeating part of said feasible two-part pre-run-time schedule."

Neither Dave, nor Dave2, nor Lindsley, nor Fong do this, as explained by the reasons given in item 49. Dave and Dave2 only use a single repeating schedule. Lindsley and Fong have no knowledge of quantitative timing constraints.

62. Allowable Subject Matter

Enclosure A: Complete Listing of All the Claims in the Application

1-114 (canceled)

115 (new): A method of scheduling on one or more processors, executions of a plurality of periodic and asynchronous processes, comprising:

(A)

automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(15) worst-case computation times for periodic processes and asynchronous processes,

(16) period for periodic processes,

(17) minimum time between two consecutive requests for asynchronous processes,

(18) deadline for periodic processes and asynchronous processes,

(19) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(20) precedence relations for periodic processes wherein each precedence relation being defined between a pair of processes comprising a first process and a second process, both said first process and said second process being periodic processes, said first process precedes said second process, execution of said

second process only allowed to start after said first process has completed its execution,

(21) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots, including the step of converting one or more asynchronous processes into corresponding new periodic processes prior to the mapping step, and mapping new periodic processes to time slots in a manner similar to mapping of other periodic processes, such that said predetermined constraints will be satisfied

(B)

during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such that said predetermined constraints will be satisfied.

116 (new): A method of scheduling on one or more processors, executions of a plurality of periodic and asynchronous processes, comprising:

(A)

automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(7) worst-case computation times for periodic processes and asynchronous processes,

(8) period for periodic processes,

(9) minimum time between two consecutive requests for asynchronous processes,

(10) deadline for periodic processes and asynchronous processes,

(11) permitted range of offset constraints for periodic processes wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(12) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots,

(B)

during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, such that said predetermined constraints will be satisfied.

117 (new): A method as defined in claim 115, including further executing a set of asynchronous processes that are not mapped to time slots during run-time of the processor at times which do not interfere with execution of processes mapped to time slots in the pre-run-time schedule.

118 (new): A method as defined in claim 115 including following pre-run-time scheduling and during run-time of the processor, the step of scheduling executions of a set of asynchronous processes that are not mapped to time slots and executions of periodic processes including said new periodic processes that are mapped to time slots such that said predetermined constraints are satisfied.

119 (new): A method as defined in claim 115, including scheduling, between the beginning time and end time of each of the time slots in the pre-run-time schedule reserved for execution of a corresponding periodic process, time capacity sufficient to complete execution of said corresponding periodic process and additional time capacity sufficient to complete execution of asynchronous processes that are not converted to new periodic processes and hence not mapped to time slots in the pre-run-time schedule in a manner similar to mapping of other periodic processes and have less latitude than said corresponding periodic process in meeting their respective deadlines.

120 (new): A method as defined in claim 116, including scheduling, between the beginning time and end time of each of the time slots in the pre-run-time schedule reserved for execution of a corresponding periodic process, time capacity sufficient to complete execution of said corresponding periodic process and additional time capacity sufficient to complete execution of asynchronous processes that are not converted to new periodic processes and hence not mapped to time slots in the pre-run-time schedule in a manner similar to mapping of other periodic processes and have less latitude than said corresponding periodic process in meeting their respective deadlines.

121 (new): A method as defined in claim 115, including prior to the mapping step, automatically adjusting lengths of periods of a predetermined set of periodic processes,

generating a set of reference periods, setting the length of the period of each periodic process to the length of the largest reference period that is no larger than an original period of the periodic process to form adjusted periods, and storing the adjusted periods for subsequent use in pre-run-time scheduling of executions of the periodic processes.

122 (new): A method as defined in claim 119, including prior to the mapping step, automatically adjusting lengths of periods of a predetermined set of periodic processes by generating a list of reference periods, setting the length of the period of each periodic process to the length of the largest reference period that is no larger than an original period of the periodic process to form adjusted periods, and storing the adjusted periods for subsequent use in pre-run-time scheduling of executions of the periodic processes.

123. (new) A method of determining whether an asynchronous process should or should not be converted into a new periodic process, comprising:
(1) calculating a first processor capacity that is required for the asynchronous process if left unconverted,
(2) calculating a second processor capacity which is required to be reserved for the new periodic process,
(3) determining whether the ratio of said first processor capacity to said second processor capacity exceeds a predetermined threshold value.

124 (new): A method as defined in claim 115 including, prior to generating the pre-run-time schedule, determining whether each asynchronous process should or should not be converted into a new periodic process, converting a subset of a predetermined set of asynchronous processes having worst-case computation time, deadline, minimum time between two requests constraints, which have been determined to be convertible, into a set of new periodic processes having worst-case computation time, period, deadline, permitted range of offset constraints and reducing possible timing conflicts with other periodic or asynchronous processes with less latitude in meeting their deadlines, by

taking into consideration the computation time requirements of the latter processes when determining the deadline of the new periodic process.

125 (new): A method as defined in claim 115 including, prior to generating the pre-run-time schedule, determining whether each asynchronous process should or should not be converted into a new periodic process, converting a subset of a predetermined set of asynchronous processes having a worst-case computation time, deadline, minimum time between two requests constraints which have been determined to be convertible, into a set of new periodic processes having release time, worst-case computation time, period, deadline, permitted range of offset constraints, wherein a permitted range of offset of each new periodic process being a subinterval of an interval or a full interval that begins at the earliest time that the corresponding being converted asynchronous process can make a request for execution, and ends at a time equal to the sum of the earliest time that said being converted asynchronous process can make a request for execution plus the period length of the new periodic process minus one time unit.

126 (new): A method as defined in claim 124, in which the determining step comprises
(1) calculating a first processor capacity that is required for the asynchronous process if left unconverted,
(2) calculating a second processor capacity which is required to be reserved for the new periodic process,
(3) determining whether the ratio of said first processor capacity to said second processor capacity exceeds a predetermined threshold value.

127 (new): A method as defined in claim 115, including generating the pre-run-time schedule as a feasible two-part pre-run-time-schedule for execution of periodic processes that may have non-zero offsets (a) an initial part which may be of zero length, and (b) a

repeating part having length which is equal to a least common multiple of lengths of the periods of the periodic processes,
all executions of all periodic processes within a time interval of length equal to the length of the least common multiple of the periodic process periods being included in the repeating part of the pre-run-time schedule, wherein all said predetermined constraints being satisfied for all executions of all periodic processes within said initial part and said repeating part.

128 (new): A method as defined in claim 127, including using any offset value in a permitted range of offsets of each periodic process, including any offset value in the permitted range of offsets of any new periodic process that may have been converted from an asynchronous process, to generate said feasible pre-run-time schedule.

129 (new): A method as defined in claim 128, wherein said permitted range of offsets of any new periodic process that may have been converted from an asynchronous process is a subinterval of an interval or a full interval that begins at the earliest time that the corresponding being converted asynchronous process can make a request for execution, and ends at a time equal to the sum of the earliest time that said being converted asynchronous process can make a request for execution plus the period length of the new periodic process minus one time unit.

130 (new): A method as defined in claim 127, further including the steps of
(A)
constructing a first schedule for executions of the periodic processes within an interval starting from zero and having length equal to maximum offset value plus a bounded number of times of the length of a least common multiple of the periodic process periods, conditions for determining feasibility requiring the existence of a point in said first schedule wherein starting from the latter point the schedule repeats in subschedule

interval lengths equal to a least common multiple of lengths of the periodic process periods, timing of all executions of all periodic processes within a time interval having length equal to the length of the least common multiple of the periodic process periods being included in each said repeating subschedule interval, and including satisfaction of all predetermined constraints for all executions of all periodic processes within the subschedule interval starting from time zero and ending at said point plus the length of the least common multiple of the periodic process periods in said first schedule, and checking for the first occurrence of said point in said first schedule,

(B)

generating said feasible two-part pre-run-time-schedule by

(1) using a subschedule interval starting from time zero and ending at said point in said first schedule as said initial part of said feasible two-part pre-run-time schedule, and

(2) using a subschedule interval starting from said point and ending at said point plus the length of the least common multiple of the periodic process periods in said first schedule as said repeating part of said feasible two-part pre-run-time schedule.

131 (new): A method as defined in claim 130, including using any offset value in a permitted range of offsets of each periodic process, including any offset value in the permitted range of offsets of any new periodic process that may have been converted from an asynchronous process, to generate said feasible pre-run-time schedule.

132 (new): A method as defined in claim 131, wherein said permitted range of offsets of any new periodic process that may have been converted from an asynchronous process is a subinterval of an interval or a full interval that begins at the earliest time that the corresponding being converted asynchronous process can make a request for execution, and ends at a time equal to the sum of the earliest time that said being converted asynchronous process can make a request for execution plus the period length of the new periodic process minus one time unit.

133 (new): A method as defined in claim 116, further comprising the steps of:
(a) generating feasible said pre-run-time schedule for the execution of a set of hard deadline periodic processes, and of a set of soft deadline periodic processes,
(b) assigning a criticality level and a deadline upper-limit to each soft deadline periodic process,
(c) in the case of not finding a feasible pre-run-time schedule under said constraints , identifying a soft critical set that contains soft deadline periodic processes for which modifying the deadlines of one or more processes in said soft critical set is necessary to meet the deadlines of all hard deadline processes,
(d) repeatedly selecting one process with a lowest criticality level among processes for which its criticality level has not reached the deadline upper-limit in the soft critical set in each case in which a feasible pre-run-time schedule has not been found, increasing the deadline of the selected process by an amount that does not exceed the deadline upper-limit of the selected process and repeatedly attempting to find a feasible pre-run-time schedule until either a feasible pre-run-time schedule is found or all the deadline upper-limits of processes in the soft critical set have been reached without finding a feasible schedule, and indicating the event of either the inability of finding a feasible schedule or of finding a feasible pre-run-time schedule, and indicating the critical set when unable to find a feasible schedule,
(e) recomputing worst-case response times of all hard deadline asynchronous processes after a feasible pre-run-time schedule has been found for all hard and soft deadline periodic processes, and in every event that the worst-case response time exceeds the deadline of a hard deadline asynchronous process, repeatedly selecting one process that has a lowest criticality level among all soft deadline periodic processes that contribute to the worst-case response time of the asynchronous process and increasing the deadline of the selected soft deadline periodic process until either (i) the worst-case response time of every hard deadline asynchronous process is less than or equal to its deadline or (ii) all the deadline upper limits of the particular set of soft deadline periodic processes that contribute to the worst-case response time of some hard deadline

asynchronous process that exceeds the deadline of said asynchronous process have been reached, and indicating the event of the latter events (i) or (ii) and the particular set.

134 (new): A method as defined in claim 115, further comprising:
(a) during run-time, or during a pre-run-time phase, using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to determine, for any point in time, whether there exists a possibility that immediate execution of a particular asynchronous process may cause the execution of any periodic process with less latitude in meeting a deadline of the latter periodic process as compared with latitude of meeting the deadline of said asynchronous process, to be delayed beyond a predetermined time limit, even if the periodic process is not ready for execution at said any point in time, and
(b) during run-time, delaying execution of said asynchronous process if said possibility is found to exist, even if said possibility is the only reason for delaying the execution of said asynchronous process at said any point in time, and even if the delay will cause the processor to be in an idle state for a time interval of non-zero length beginning from said any point in time.

135 (new): A method as defined in claim 115, further comprising:
(a) either during run-time, or during a pre-run-time phase, using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to determine, for any point in time, whether there exists a possibility that immediate execution of a particular asynchronous process may cause the execution of any periodic process with less latitude in meeting a deadline of the latter periodic process as compared with latitude of meeting the deadline of said asynchronous process, to be delayed beyond the end of the time slot of the periodic process in the pre-run-time schedule, even if the periodic process is not ready for execution at said any point in time, and

(b) during run-time, delaying execution of said asynchronous process if said possibility is found to exist, even if said possibility is the only reason for delaying the execution of said asynchronous process at said any point in time, and even if the delay will cause the processor to be in an idle state for a time interval of non-zero length beginning from said any point in time.

136 (new): A method as defined in claim 115, further comprising:
(a) either during run-time, or during a pre-run-time phase, using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to determine, for any point in time, whether there exists a possibility that immediate execution of a particular asynchronous process may cause the execution of said asynchronous process, or the execution of some other asynchronous process, to extend beyond the beginning of the time slot of any periodic process that has not yet started in the pre-run-time schedule, even if the periodic process is not ready for execution at said any point in time, and
(b) during run-time, delaying execution of said asynchronous process if said possibility is found to exist, even if said possibility is the only reason for delaying the execution of said asynchronous process at said any point in time, and even if the delay will cause the processor to be in an idle state for a time interval of non-zero length beginning from said any point in time.

137 (new): A method as defined in claim 115, further comprising:
(a) either during run-time, or during a pre-run-time phase, using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to determine, for any point in time, whether there exists a possibility that immediate execution of a particular asynchronous process may cause the execution of any periodic process to be delayed beyond the end of the time slot of that periodic process in the pre-run-time schedule, even if the periodic process is not ready for execution at said any point in time, and

(b) during run-time, delaying execution of said asynchronous process if said possibility is found to exist, even if said possibility is the only reason for delaying the execution of said asynchronous process at said any point in time, and even if the delay will cause the processor to be in an idle state for a time interval of non-zero length beginning from said any point in time.

138 (new): A method as defined in claim 115, further comprising:
(a) either during run-time, or during a pre-run-time phase, using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to determine, for any point in time, whether there exists a possibility that immediate execution of a particular first asynchronous process may cause execution of any second asynchronous process to be continuously blocked for a duration of the execution of the first asynchronous process and a duration of the execution of any periodic process, when the second asynchronous process has less latitude in meeting the deadline of the second asynchronous process as compared with the latitude of both the first asynchronous process in meeting the deadline of the first asynchronous process and the latitude of the periodic process in meeting the deadline of the periodic process, even if neither the periodic process nor the second asynchronous process are ready for execution at said any point in time, and
(b) during run-time, delaying execution of the first asynchronous process if said possibility is found to exist, even if said possibility is the only reason for delaying the execution of the first asynchronous process at said point in time, and even if the delay will cause the processor to be in an idle state for a time interval of non-zero length beginning from said any point in time.

139 (new): A method as defined in claim 116, comprising during run-time, detecting, in a case in which no asynchronous process or periodic process that has started is to be immediately put into execution, conditions of whether there exists an execution of a first periodic process that is ready for execution and has not completed execution, and there

does not exist any other execution of some second periodic process that has not yet completed, such that execution of the second periodic process is ordered before execution of the first periodic process in the pre-run-time schedule[[,]] and the time slot of the first periodic process is not nested within the time slot of the second periodic process in the pre-run-time schedule, and there does not exist any other execution of some third periodic process that is ready and has not completed execution, such that execution of the third periodic process is nested within the time slot of the first periodic process in the pre-run-time schedule, and beginning execution of the first periodic process immediately in the event said conditions are true.

140 (new): A method as defined in claim 115, including determining a worst-case response time of an asynchronous process that has not been converted into a periodic process using a formula comprising:
the sum of worst-case computation times of asynchronous processes and periodic processes that have less or equal latitude as compared with the latitude of the asynchronous process in meeting their respective deadlines,
plus the maximum time that the asynchronous process may possibly be blocked by some asynchronous or periodic process that has greater latitude as compared with the latitude of the asynchronous process in meeting their respective deadlines, plus the worst-case computation time of the asynchronous process multiplied by the number of periodic processes with which the asynchronous process has an exclusion relation.

141 (new): A method as defined in claim 115, including, during a pre-run-time phase, determining by simulation a worst-case response time of an asynchronous process corresponding to a feasible pre-run-time schedule of periodic processes consisting of an initial part of the pre-run-time schedule which may be of zero length and a repeating part of the pre-run-time schedule, wherein for each point in time from zero to the end time of the repeating part of the run-time schedule minus one time unit, simulating execution of the asynchronous process using functions used to determine a run-time schedule and

recording response time of the asynchronous process under the assumption that the asynchronous process arrives at a point in time under consideration, all other asynchronous processes that can possibly block the asynchronous process arriving at one time unit prior to the point in time under consideration, and all asynchronous processes that have less latitude than latitude of the asynchronous process arriving at the same said point of time under consideration, scheduling executions of periodic processes to start at the beginning time and to complete execution at the end time of their respective time slots in the pre-run-time schedule, wherein whenever the asynchronous process is delayed because it may block execution of some periodic process having less latitude than the latitude of the asynchronous process, or may block execution of some second asynchronous process for the duration of more than one execution of processes having greater latitude as compared with the latitude of the second asynchronous process, all asynchronous
processes having less latitude as compared with the latitude of the asynchronous process is delayed in order to delay the asynchronous process for a maximum possible amount of time, thus simulating all possible worst-case scenarios of executions of the asynchronous process.

142 (new): A method as defined in claim 115, including restricting every periodic process in the pre-run-time schedule to be executed strictly within its time slot.

143 (new): A method as defined in claim 115, including, during a pre-run-time phase, generating tables of safe start time intervals for the executions of asynchronous processes, wherein every periodic process in the pre-run-time schedule is scheduled to be executed strictly within its time slot, wherein for every point in time of the pre-run-time schedule, it is determined whether each asynchronous process should be delayed, under the assumption that the actual start time of execution of every periodic process is equal to the beginning time of its time slot, and the actual end time of execution of every periodic process is equal to the end time of its time slot, wherein for every point in time of the pre-

run-time schedule, in the event said asynchronous process is to be delayed according to the assumptions, the point in time is set to be unsafe and recorded in a corresponding entry in the table for the point in time and said asynchronous process.

144 (new): A method as defined in claim 115, including, during a pre-run-time phase, generating tables of safe start time intervals for the executions of asynchronous processes, wherein every periodic process is scheduled to be executed strictly within its time slot in the pre-run-time schedule,
wherein for selected points in time of the pre-run-time schedule, it is determined whether each asynchronous process should be delayed, under the assumption that the actual start time of execution of every periodic process is equal to the beginning time of its time slot, and the actual end time of execution of every periodic process is equal to the end time of its time slot, wherein for selected points in time of the pre-run-time schedule, in the event said asynchronous process is to be delayed according to the assumptions, that point in time is set to be unsafe and recorded in a corresponding entry in the table for the point in time and said asynchronous process.

145 (new): A method as defined in claim 115 comprising during run-time, detecting at least one event of, at any point in time, whether some asynchronous process has arrived by said point in time, whether some asynchronous process or periodic process has completed its computation at said point in time, and whether said point in time is both the release time and beginning time of a time slot in the pre-run-time schedule for some periodic process, and activating a run-time scheduler at said point in time, and should said at least one event have occurred, determining whether any asynchronous process that has arrived but has not yet been completed should be delayed or immediately put into execution, and including the further step of delaying execution of a first asynchronous process when execution of some other asynchronous process that excludes a periodic process with less or equal latitude as compared with the latitude of the first asynchronous

process in meeting their respective deadlines has already started but has not yet been completed.

146 (new): A method as defined in claim 116 comprising during run-time, detecting at least one event of, at any point in time, whether some asynchronous process has arrived by said point in time, whether some asynchronous process or periodic process has completed its computation at said point in time, and whether said point in time is both the release time and beginning time of a time slot in the pre-run-time schedule for some periodic process, and activating a run-time scheduler at said point in time, and should said at least one event have occurred, determining whether any asynchronous process that has arrived but has not yet been completed should be delayed or immediately put into execution, and including the further step of delaying execution of an asynchronous process in the event there exists the possibility that immediate execution of the latter asynchronous process may cause execution of a first periodic process with less or equal latitude to be delayed, when execution of the first periodic process may be preempted by execution of some second periodic process, and the latter asynchronous process cannot be preempted by the second periodic process.

147 (new): A method as defined in claim 116 comprising during run-time, detecting at least one event of, at any point in time, whether some asynchronous process has arrived by said point in time, whether some asynchronous process or periodic process has completed its computation at said point in time, and whether said point in time is both the release time and beginning time of a time slot in the pre-run-time schedule for some periodic process, and activating a run-time scheduler at said point in time, and should said at least one event have occurred, determining whether any asynchronous process that has arrived but has not yet been completed should be delayed or immediately put into execution, and including the further step of delaying execution of an asynchronous process when it is not allowed to preempt execution of any first process that has already started and has not completed execution and that excludes some other second

asynchronous process which has latitude that is less than both said any first process and the latitude of said second asynchronous process, whereby blocking of the second asynchronous process by the duration of more than one execution of processes with greater latitude thereof may be avoided.

148 (new): A method as defined in claim 116 comprising during run-time, detecting at least one event of, at any point in time, whether some asynchronous process has arrived by said point in time, whether some asynchronous process or periodic process has completed its computation at said point in time, and whether said point in time is both the release time and beginning time of a time slot in the pre-run-time schedule for some periodic process, and activating a run-time scheduler at said point in time, and should said at least one event have occurred, determining whether any asynchronous process that has arrived but has not yet been completed should be delayed or immediately put into execution, and including the further step of delaying execution of an asynchronous process to allow preemption of execution of the asynchronous process by execution of a first periodic process that is ready for execution and that has latitude less than or equal to the latitude of the asynchronous process, when the asynchronous process does not exclude the first periodic process and does not exclude any other asynchronous process with a latitude that is less than the latitude of the first periodic process, and there does not exist execution of some second periodic process that has not been completed such that execution of the second periodic process is ordered before execution of the first periodic process and execution of the time slot of the first periodic process is not nested within the time slot of the second periodic process in the pre-run-time schedule.

149 (new): A method as defined in claim 115, including, during a pre-run-time phase, determining by simulation a worst-case response time of an asynchronous process corresponding to a feasible pre-run-time schedule of periodic processes, wherein for each point in time from zero to the end time of the repeating part of the run-time schedule minus one time unit, simulating execution of said asynchronous process using functions

used to determine a run-time schedule and recording response time of said asynchronous process under the assumption that said asynchronous process arrives at a point in time under consideration, all other asynchronous processes that can possibly block said asynchronous process arriving at one time unit prior to the point in time under consideration, and all asynchronous processes that have less latitude than latitude of said asynchronous process arriving at the same said point of time under consideration, scheduling executions of periodic processes to start at the beginning time and to complete execution at the end time of their respective time slots in the pre-run-time schedule, simulating all possible worst-case scenarios of executions of said asynchronous process.

150 (new): A method of scheduling on one or more processors, executions of a plurality of periodic and asynchronous processes, comprising:

(A)

automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(1) worst-case computation times for periodic processes and asynchronous processes,

(2) period for periodic processes,

(3) minimum time between two consecutive requests for asynchronous processes,

(4) deadline for periodic processes and asynchronous processes,

(5) precedence relations for periodic processes wherein each precedence relation being defined between a pair of processes comprising a first process and a second process, both said first process and said second process being periodic processes, said first process precedes said second process, execution of said second process only allowed to start after said first process has completed its execution,

(6) exclusion relations for periodic and asynchronous processes wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process being either a periodic process or an asynchronous process and said second process being either a periodic process or an asynchronous process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots, including scheduling, between the beginning time and end time of each of the time slots in the pre-run-time schedule reserved for execution of a corresponding periodic process, time capacity sufficient to complete execution of said corresponding periodic process and additional time capacity sufficient to complete execution of asynchronous processes that are not converted to new periodic processes and hence not mapped to time slots in the pre-run-time schedule in a manner similar to mapping of other periodic processes and have less latitude than said corresponding periodic process in meeting their respective deadlines.

(B)

during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions, including allowing executions of asynchronous processes that have not been mapped to time slots in the pre-run-time schedule to be completed within any time slot of a periodic process that has greater latitude in meeting its deadline, such that said predetermined constraints will be satisfied.

151 (new): A method as defined in claim 150 including, prior to the mapping step, converting one or more asynchronous processes having a worst-case computation time, deadline, minimum time between two requests constraints, into a set of new periodic processes, and reducing possible timing conflicts with other periodic or asynchronous

processes with less latitude in meeting their deadlines, by taking into consideration the computation time requirements of the latter processes when determining the deadline of each of the new periodic processes, and mapping the new periodic process in a manner similar to mapping of other periodic processes.

152 (new): A method as defined in claim 151, in which the determining step is performed by calculating whether a ratio of processing capacity of the processor which is required to be reserved for new periodic processes, to processor capacity that is required for the asynchronous process if left unconverted, exceeds a predetermined threshold value.

153 (new): A method of scheduling on one or more processors, executions of a plurality of periodic processes, comprising:

(A)

automatically generating a pre-run-time schedule comprising mapping from a set of periodic process executions to a sequence of time slots on one or more processor time axes, each of the time slots having a beginning time and an end time, reserving each one of the time slots for execution of one of the periodic processes, the positions of the end time and the beginning time of each of the time slots being such that execution of the periodic processes,

including satisfaction of predetermined constraints comprising

(7) worst-case computation times,

(8) period,

(9) deadline,

(10) permitted range of offset constraints wherein a permitted range of offset of a periodic process comprising an interval that begins at a lower bound value and ends at an upper bound value which may be equal to the lower bound value, the duration of the time interval between the beginning of the first period of said periodic process and time zero must be greater than or equal to said lower bound value and less than or equal to said upper bound value,

(11) precedence relations wherein each precedence relation being defined between a pair of processes comprising a first process and a second process, execution of said second process only allowed to start after said first process has completed its execution,

(12) exclusion relations wherein each exclusion relation being defined between a pair of processes comprising a first process and a second process, said first process excludes said second process, no execution of said second process being allowed to occur between the time that said first process starts its execution and the time that said first process completes its computation,

can be completed between the beginning time and end time of respective time slots,

(B)

during run-time using the information in the pre-run-time schedule, including the positions of the beginning time and end time of the time slots of the periodic processes, to schedule the process executions such that said predetermined constraints will be satisfied.

154 (new): A method as defined in claim 153, including generating the pre-run-time schedule as a feasible two-part pre-run-time-schedule for execution of periodic processes that may have non-zero offsets (a) an initial part which may be of zero length, and (b) a repeating part having length which is equal to a least common multiple of lengths of the periods of the periodic processes,

all executions of all periodic processes within a time interval of length equal to the length of the least common multiple of the periodic process periods being included in the repeating part of the pre-run-time schedule, wherein all said predetermined constraints being satisfied for all executions of all periodic processes within said initial part and said repeating part.

155 (new): A method of determining whether each asynchronous process in an original set of asynchronous processes should or should not be converted into a new periodic

process and included in a set of periodic processes that will be mapped to time slots in a pre-run-time schedule,

each asynchronous process in said original set of asynchronous processes has predetermined asynchronous process constraints comprising worst-case computation time, deadline, and minimum time between two consecutive requests constraints,

each periodic process in said set of periodic processes has predetermined periodic process constraints comprising permitted range of offset, worst-case computation time, deadline, period constraints,

each new periodic process has new periodic process constraints comprising permitted range of offset, worst-case computation time, deadline, period constraints,

said method steps comprising:

(A)

selecting one asynchronous process in said original set of asynchronous processes,

(1) tentatively converting said one asynchronous process into a corresponding new periodic process, such that said asynchronous process constraints of said one asynchronous process will be satisfied by said new periodic process constraints of said corresponding new periodic process,

(2) calculating a first processor capacity which is required to be reserved if said one asynchronous process is not converted,

(3) calculating a second processor capacity which is required to be reserved if said one asynchronous process is converted,

(4) including a copy of said one asynchronous process in a second set of asynchronous processes if the ratio of said first processor capacity to said second processor capacity exceeds a predetermined threshold, otherwise including said tentatively converted corresponding new periodic process in said set of periodic processes,

(B)

repeating A until every asynchronous processes in said original set of asynchronous processes has been selected,

(C)

repeating B,

including removing from the periodic set any corresponding new periodic process that was included in the periodic set in any previous execution of B whenever a copy of a asynchronous process is to be included in the second set of asynchronous processes in the current execution of step B,
further including removing any copy of an asynchronous process from the second asynchronous set that was included in the second asynchronous set in any previous execution of step A whenever any corresponding new periodic process is to be included in the set of periodic processes in the current execution of step A,

(D)

terminating when no more changes to the second asynchronous set have occurred between the beginning and the end of any execution of C.

156 (new): A method as defined in claim 124, the determining step comprises
(1) calculating a first processor capacity that is required for the asynchronous process if left unconverted,
(2) calculating a second processor capacity which is required to be reserved for the new periodic process,
(3) determining whether the ratio of said first processor capacity to said second processor capacity exceeds a predetermined threshold value."

157 (new): A method as defined in claim 154, further including the steps of
(A)
constructing a first schedule for executions of the periodic processes within an interval starting from zero and having length equal to maximum offset value plus a bounded number of times of the length of a least common multiple of the periodic process periods, conditions for determining feasibility requiring the existence of a point in said first schedule wherein starting from the latter point the schedule repeats in subschedule interval lengths equal to a least common multiple of lengths of the periodic process periods, timing of all executions of all periodic processes within a time interval having

length equal to the length of the least common multiple of the periodic process periods being included in each said repeating subschedule interval, and including satisfaction of all predetermined constraints for all executions of all periodic processes within the subschedule interval starting from time zero and ending at said point plus the length of the least common multiple of the periodic process periods in said first schedule, and checking for the first occurrence of said point in said first schedule,

(B)

generating said feasible two-part pre-run-time-schedule by

(1) using a subschedule interval starting from time zero and ending at said point in said first schedule as said initial part of said feasible two-part pre-run-time schedule, and

(2) using a subschedule interval starting from said point and ending at said point plus the length of the least common multiple of the periodic process periods in said first schedule as said repeating part of said feasible two-part pre-run-time schedule.